

Uwe Kloos, Natividad Martínez, Gabriela Tullius (Hrsg.)

Informatics Inside:

Reality++
Tomorrow comes today!



Tagungsband





Uwe Kloos, Natividad Martínez, Gabriela Tullius (Hrsg.)

Informatics Inside:

Reality++
Tomorrow comes today!

Informatik-Konferenz an der Hochschule Reutlingen
25. April 2012

ISBN 978-3-00-037938-3



INF

Fakultät
Informatik



Impressum

Anschrift:
Hochschule Reutlingen
Reutlingen University
Fakultät Informatik
Medien- und Kommunikationsinformatik
Alteburgstraße 150
D-72762 Reutlingen

Telefon: +49 (0)7121 / 271-4002
Telefax: +49 (0)7121 / 271-4042

E-Mail: infoinside@reutlingen-university.de
Internet: http://www.infoinside.reutlingen-university.de

Organisationskomitee:
Prof. Dr. Uwe Kloos, Hochschule Reutlingen
Prof. Dr. Natividad Martínez, Hochschule Reutlingen
Dipl.-Betriebswirtin (BA) Danja Walz, Hochschule Reutlingen

Patrick Gaßner
Frank Griesinger
Erik Herrmann
Johannes Hihn
Thomas Hornstein
Samet Keser
Marita Klein
Christoph Meißner
Michael Schempp
Hannah-Elena Seeger

Herstellung und Verlag: Hochschule Reutlingen
ISBN 978-3-00-037938-3



Vorwort

Bereits zum vierten Mal findet nun die Informatics Inside an der Hochschule Reutlingen statt. Diese wissenschaftliche Konferenz des Masterstudiengangs Medien- und Kommunikationsinformatik wird von den Studierenden organisiert und durchgeführt. Sie erhalten während ihres Masterstudiums die Gelegenheit sich in einem Fachgebiet ihrer Wahl zu vertiefen. Dies kann an der Hochschule, in einem Unternehmen, einem Forschungsinstitut oder im Ausland durchgeführt werden. Gerade diese flexible Ausgestaltung der Lehrveranstaltung „Wissenschaftliche Vertiefung" führt zu vielfältigen Themenfelder, die von den Studierenden bearbeitet werden. Neben der eigentlichen fachlichen Vertiefung spielt auch die Präsentation von wissenschaftlichen Ergebnissen eine wichtige Rolle und dies nicht nur während des Studiums. Ein gewähltes Fachgebiet so allgemeinverständlich aufzubereiten und zu vermitteln, dass es auch für Nicht-Spezialisten verständlich wird, stellt immer wieder eine besondere Herausforderung dar. Dieser Herausforderung stellen sich die Studierenden im Rahmen der Konferenz zur wissenschaftlichen Vertiefung am 25. April 2012. Erstmalig wird die Konferenz dabei ohne zusätzliche Firmenausstellung stattfinden. Die Firmen selber werden sich am folgenden Donnerstag, dem 26. April im Rahmen einer hochschulweiten Firmenmesse vorstellen.

Das Motto der diesjährigen Konferenz lautet „Reality++: Tomorrow comes today!". Unter diesem fast schon visionären Thema werden die Ergebnisse der verschiedenen Vertiefungsarbeiten aus den vergangenen Monaten präsentiert. Sie werden in mehreren Fachvorträgen einen Einblick in die unterschiedlichen Aktivitäten der Studierenden erhalten. Das Programm wird vervollständigt durch Beiträge von Experten aus der Forschung und Industrie. Mit diesem Tagungsband halten Sie die schriftliche Ausarbeitung der Fachbeiträge in der Hand und können sich selber ein Bild von den Arbeiten der Studierenden machen. Weiterhin erhalten Sie Einblicke in verschiedene Projekte der Fakultät Informatik, insbesondere aus dem Studiengang Medien- und Kommunikationsinformatik. Ich wünsche Ihnen eine spannende Lektüre, die Sie vielleicht mit einigen besonderen Ergebnissen überraschen wird.

Prof. Dr. Uwe Kloos
Dekan der Fakultät Informatik

Inhaltsverzeichnis



XML3D Kinect Support: Webbrowser-Plug-in und Navigationsanwendung

Erik Herrmann
Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Erik.Herrmann@Student.Reutlingen-University.de

ABSTRACT

JavaScript-Engines moderner Webbrowser stellen plattformunabhängige und leistungsfähige Laufzeitumgebungen dar, die durch WebGL auch für 3D-Anwendungen genutzt werden können. XML3D hat das Ziel, die Entwicklung dieser Anwendungen zu vereinfachen und stellt einen Standardisierungsvorschlag des DFKI für eine Markup-Sprache zur Beschreibung von 3D-Szenen dar. Um Entwickler nicht nur auf die Eingabemöglichkeiten einzuschränken, die durch die Webbrowser unterstützt werden, werden dabei innerhalb eines Teilprojektes Schnittstellen zu alternativen Eingabegeräten entwickelt. Dieser Artikel beschreibt die Umsetzung einer in diesem Rahmen entwickelten Schnittstelle zu einem Skelett-Tracker, basierend auf dem Kinect-Sensor. Die Verwendbarkeit der Schnittstelle wurde mit einer Gesten-basierten Navigationsanwendung für XML3D-Szenen getestet.

1 EINLEITUNG

JavaScript-Engines moderner Webbrowser stellen plattformunabhängige und leistungsfähige Laufzeitumgebungen dar, die durch HTML5 und der WebGL-API[1] auch für die Entwicklung von 3D-Anwendungen genutzt werden können. XML3D[2], entwickelt durch das DFKI[3], hat das Ziel, die Entwicklung dieser plattformunabhängigen Anwendungen zu vereinfachen und stellt einen Standardisierungsvorschlag einer Markup-Sprache zur Beschreibung von 3D-Szenen dar. Um Entwickler nicht nur auf die Eingabemöglichkeiten einzuschränken, die durch die Webbrowser unterstützt werden, werden dabei innerhalb eines Teilprojektes Schnittstellen zu alternativen Eingabegeräten entwickelt. In diesem Artikel wird ein in diesem Rahmen entwickeltes Webbrowser-Plug-in vorgestellt, das eine JavaScript-Schnittstelle zu Skelett-Tracker-Frameworks für den Kinect-Sensor von Microsoft darstellt. Ursprünglich entwickelt als Eingabegerät für Unterhaltungselektronik, stellt dieser einen kostengünstigen Tiefensensor für Endnutzer dar, für den Frameworks vorhanden sind, die u.a. das markerlose Tracking der Gelenkpositionen von bis zu zwei Personen bieten. Bei der Entwicklung des Plug-ins wurden OpenNI/NITE[4] und die Beta des Microsoft Kinect SDKs[5] betrachtet, die um die Erkennung von Handgesten erweitert wurden. Die Leistungsfähigkeit des Plug-ins wurde zudem mit einer Gesten-basierten Navigationsanwendung für XML3D-Szenen getestet, die in JavaScript implementiert wurde.

2 WEBBROWSER-PLUG-IN

Die Problemstellung bestand darin, einer JavaScript-Anwendung im Webbrowser einen konstanten Strom von Trackingdaten mehrerer Benutzer zur Verfügung zu stellen. Bei dem Skelett-Tracker von OpenNI/NITE bestehen diese Daten aus Koordinaten und der Orientierung von 24 Gelenken von bis zu zwei Benutzern und bei dem Microsoft Kinect SDK aus Koordinaten von 20 Gelenken von ebenfalls bis zu zwei Benutzern.

Als Lösung der Problemstellung entschied man sich, je Framework ein sog. skriptbares

---

[1] http://www.khronos.org/webgl/
[2] http://www.xml3d.org
[3] Deutsches Forschungsinstitut für künstliche Intelligenz: http://www.dfki.de/web
[4] http://openni.org
[5] http://kinectforwindows.org

NPAPI-Plug-in (Netscape Plug-in API) zu entwickeln, das als Wrapper dienen soll. Bei NPAPI [4] handelt es sich um eine standardisierte Plug-in-Schnittstelle, die von den meisten Webbrowsern unterstützt wird. Diese Plug-ins ermöglichen die Darstellung von Datenformaten innerhalb eines Dokuments, die dem Webbrowser selbst unbekannt sind, wie z.B. PDF- und Flash-Dateien. Zudem existiert durch die Erweiterung „npruntime“ die Möglichkeit der Entwicklung von sog. skriptbaren Plug-ins, die nativ in C++ eine Klasse implementieren, die von der JavaScript-Engine als Objekt instanziiert werden kann. Die nativ implementierten Methoden dieses Objekts können so von JavaScript aus aufgerufen werden. Bei den Attributen dieses Objekts kann es sich zudem um Instanzen anderer vom Plug-in implementierter Klassen handeln, wodurch vom Plug-in eine skriptbare Objektstruktur aufgebaut werden kann. Zusammen mit der Möglichkeit vom Plug-in aus JavaScript-Funktionen aufzurufen, stellen NPAPI-Plug-ins so einen idealen Wrapper dar, weshalb die Alternativen, z.B. die Übertragung der Daten über das Netzwerk, hier nicht betrachtet werden.

2.1 SCHNITTSTELLE

Die Hauptaufgabe der umgesetzten Plug-ins besteht in der Initialisierung des jeweiligen Frameworks und dem Starten eines Threads, der bei jeder Aktualisierung des Skelett-Trackers die gelieferten Daten in eine skriptbare Objektstruktur schreibt. Zur Interpretation der gelieferten Daten durch eine JavaScript-Anwendung, ermöglicht das Plug-in die Registrierung einer JavaScript-Funktion als Event-Handler, der bei jeder Aktualisierung

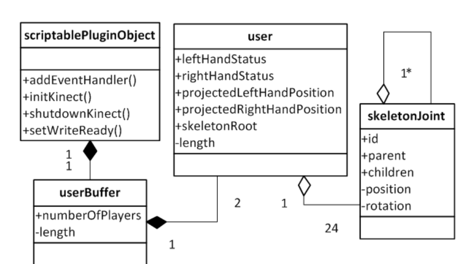


Abbildung 1: Die skriptbare Plug-in-Schnittstelle

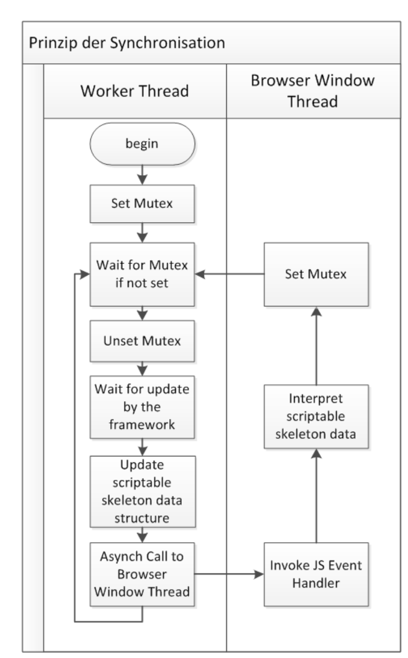


Abbildung 2: Synchronisation zwischen JavaScript-Anwendung und Worker-Thread

benachrichtigt wird. Die identische Schnittstelle beider Plug-ins ist in Abbildung 1 dargestellt.

Da NPAPI nicht Thread-sicher ist, können NPAPI-Funktionen, z.B. zum Aufruf des JavaScript-Event-Handlers, nur von dem Thread des Browser-Fensters verwendet werden, mit dem das Plug-in initialisiert wurde. Zur Lösung des Problems, bietet NPAPI jedoch die Möglichkeit eine Nachricht von Worker-Threads an die sog. Message Pump des Browser-Threads zu schicken, um asynchron darüber einen Funktionsaufruf zu initiieren. Um den Worker-Thread mit der JavaScript-Anwendung nach dem asynchronen Aufruf wieder zu synchronisieren, wartet dieser bis der JavaScript-Event-Handler signalisiert, dass die Daten interpretiert wurden. Dieser Ablauf ist in Abbildung 2 dargestellt.

Bei ersten Testanwendungen, deren Event-Handler in einer hohen Frequenz Daten aus der skriptbaren Objektstruktur gelesen haben, wurde ein problematischer Anstieg der Speicherbe-

legung beobachtet. Die Ursache hierfür liegt darin, dass die JavaScript-Engine bei dem Zugriff auf ein Attribut des skriptbaren Plug-ins eine typunsichere JavaScript-Variable erstellt, in die der zurückgegebene Wert geschrieben wird. Es wird vermutet dass der allokierte Speicher von Variablen mit einer Referenz auf ein natives Objekt aus der skriptbaren Objektstruktur von dem Garbage-Collector jedoch nicht mehr freigegeben werden kann, da die Bedingung hierfür, dass der Referenzzähler des referenzierten Objekts Null erreicht, erst bei dem Schließen des Browserfensters erfüllt wird. Zur Lösung des Problems kann jedoch während der Initialisierung einer Anwendung einmalig eine Referenz auf jedes benötigte skriptbare Objekt in einer statischen Variable gespeichert und daraufhin von dem Event-Handler wiederverwendet werden, ohne dass es zum Speicheranstieg kommt.

2.2 HANDGESTEN

Bei der Entwicklung von Gesten-basierten Anwendungen können bekannte Gesten zum Bestätigen von Eingaben wiederverwendet werden. Aus diesem Grund hat man sich entschieden das Plug-in um solche Gesten zu erweitern. OpenNI/NITE bietet zu diesem Zweck die Erkennung von Handgesten über die NITE-Toolbox-Bibliothek. Diese Bibliothek wurde jedoch ignoriert, da sie zum Entwicklungszeitpunkt auf dem Hand-Tracker von OpenNI/NITE basierte und eine eigene Initialisierungsgeste für jede Hand erforderte. Desweiteren wäre diese Lösung inkompatibel mit dem Microsoft Kinect SDK gewesen. Aus diesem Grund wurde die Erkennung für zwei Handgesten, eine Drück- und eine Greifgeste, selbst implementiert. Zur Verwendung dieser Gesten können über die skriptbare Schnittstelle Event-Handler registriert werden.

Bei der Erkennung der Drückgeste werden die Handkoordinaten auf schnelle Bewegungen in Richtung der Kinect überwacht. Hierzu verfügt jeder Benutzer für jede Hand über eine Bewegungshistorie der letzten acht Frames, mit der die durchschnittliche Änderung der Koordinaten berechnet wird. Damit die Bewegung korrekt interpretiert wird, muss die Z-Achse des Skelett-Koordinatensystems parallel zum Boden ausgerichtet sein. Aufgrund der erforderten schnellen Bewegung der Hand, war mit der implementierten Drückgeste jedoch keine präzise Selektion möglich, weshalb zudem die Greifgeste entwickelt wurde. Diese Geste erlaubt die Hand ruhig zu halten, da sie auf der Erkennung eines Zustandswechsels der Hand von offen auf geschlossen und wieder auf offen basiert. Zum Zeitpunkt der Entwicklung boten keine der beiden Frameworks eine hierfür notwendige Unterscheidung offener und geschlossener Hände, weshalb diese selbst implementiert wurde (siehe Kapitel 2.3). Um unbewusste Zustandswechsel von der Greifgeste zu unterscheiden, wurde eine minimale und maximale Wartezeit zwischen dem Schließen und dem Öffnen der Hand eingeführt. Die Wartezeit wurde dabei auf minimal 450 und maximal 1500 Millisekunden gesetzt. Um eine visuelle Rückmeldung für jeden auftretenden Zustandswechsel der Greifgeste geben zu können, können dafür Event-Handler registriert werden.

2.3 UNTERSCHEIDUNG DES HANDZUSTANDS

Für die Umsetzung der Greifgeste, war es nötig eine Unterscheidung von geschlossenen und geöffneten Händen zu implementieren. Hierbei hat man sich an den Ansatz vorgestellt von Manresa et al. [3] orientiert. Dieser funktioniert durch das Erkennen der Tiefe von Defekten in der konvexen Hülle der Kontur einer Hand. Ein Defekt in der konvexen Hülle entspricht hierbei einer Lücke zwischen zwei Fingern und wird durch einen Startpunkt u, einer Tiefe d und einen Endpunkt v definiert (siehe Abbildung 3).

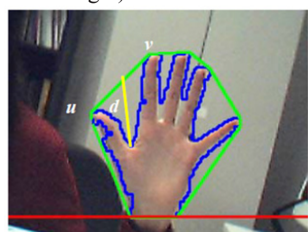


Abbildung 3: Defekte in einer konvexen Hülle [3]

Basierend auf der Anzahl und der Tiefe der gefunden Defekte im Vergleich zu einem empirisch ermittelten Grenzwert kann somit erkannt werden, ob eine Hand offen oder geschlossen ist.
Die Funktionen zum Finden einer Kontur, deren konvexe Hülle und deren Defekte in einem binären Bild sind in OpenCV[6] implementiert und von Bradski et al. [2] dokumentiert, wodurch der Algorithmus einfach auf ein Tiefenbild übertragen werden konnte. Die Implementierung wurde zudem vereinfacht, da die Positionen der Hände bereits durch den Skelett-Tracker geliefert werden. Zur Lösung des Problems, dass die gelieferten Handpositionen bei schnellen Bewegungen nicht mit dem Tiefenbild übereinstimmen, überprüft das Plug-in den Handzustand nur unterhalb einer Höchstgeschwindigkeit. Da sich Benutzer auch in verschiedenen Entfernungen zum Kinect-Sensor aufhalten können, kann die Auflösung einer Hand im Tiefenbild und folglich die Tiefe der Defekte variieren. Für die korrekte Unterscheidung der Handzustände muss deshalb der Grenzwert für die Prüfung der Konvexitätsdefekte relativ zu dem Tiefenwert der jeweiligen Hand berechnet werden. Um die Zustände der Hände auch über diese Geste hinaus z.B. für die implementierte Navigationsanwendung zu nutzen, sind diese über die skriptbare Plug-in-Schnittstelle abfragbar.

3 NAVIGATIONSANWENDUNG

Zur Demonstration der Lauffähigkeit des Plug-ins, wurde eine Navigationsanwendung für XML3D-Szenen umgesetzt. Ziel der Anwendung war die Erkundung von Szenen durch eine direkte Kontrolle der Position und Orientierung der Kamera. Hierbei hat man sich an den Techniken orientiert, die von Bowman et al. [1] beschrieben werden, wonach Lenkungstechniken den höchsten Grad an Kontrolle über den virtuellen Standpunkt bieten. Da sich die Selektion von Menüeinträgen einfach auf die Hände eines Benutzers abbilden lässt, hat man sich zudem entschieden eine zielbasierte Navigation über ein Menü bereitzustellen, über das der virtuelle Standpunkt gespeichert und wiederhergestellt werden kann. Da man während erster Tests ein robusteres Tacking der Handpositionen durch OpenNI/NITE beobachtet hatte, fand die Entwicklung der Anwendung mit diesem Framework statt. Nach der Fertigstellung wurde sie auf das Koordinatensystem des Microsoft Kinect SDKs angepasst.

[6] http://opencv.willowgarage.com/wiki

3.1 LENKUNGSTECHNIK

Da der Benutzer stationär vor dem Monitor und der Kinect bleiben muss, um von dem Skelett-Tracker erfasst zu werden, konnte keine der von Bowman et al. beschriebenen Lenkungstechniken für immersive Displays und Marker-basierte Trackingsysteme direkt auf die Kinect übertragen werden. Aus diesem Grund wurde eine virtuelle Steuerung mit einer relativen Berechnung der Translation und der Orientierung der Kamera auf die Hände des Benutzers abgebildet. Diese Lenkungstechnik wird aktiviert, wenn beide Hände geschlossen und beide Unterarme parallel zueinander auf die Kinect ausgerichtet sind (siehe Abbildung 4).

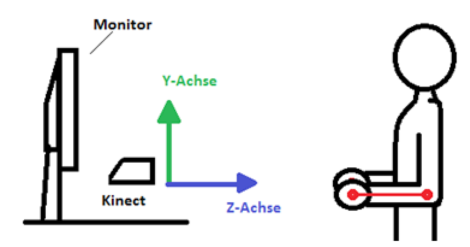


Abbildung 4: Initialisierungspose der Lenkungstechnik

Die Koordinaten beider Hände werden dabei gespeichert und dienen als Ausgangspunkte für die Berechnung der Translation und Orientierung der Kamera. Die Kontrolle bleibt daraufhin aktiv bis beide Hände geöffnet werden. Abbildung 5 zeigt die Rolle beider Hände während der Navigation und die korrespondierende visuelle Rückmeldung gezeichnet auf einem HTML5-Canvas-Element, das über dem HTML5-Canvas-Element angeordnet wird, das die Szene darstellt. Da eine haptische Rückmeldung fehlt, nimmt diese Visualisierung eine wichtige Rolle bei der Navigation ein.

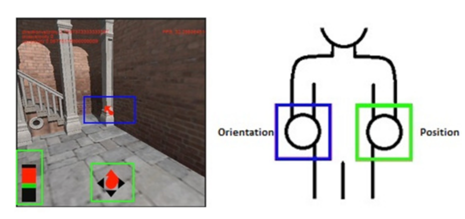


Abbildung 5: Abbildung der relativen Handbewegung auf die Kontrolle der Kamera und der visuellen Rückmeldung

Während der aktiven Navigation wird die Differenz der Koordinaten der linken Hand zu deren Ausgangsposition auf die Translation abgebildet und die Differenz der Koordinaten der rechten Hand zu deren Ausgangsposition auf die Orientierung.
So ist durch Bewegungen der linken Hand entlang der Z-, X- und Y-Achse des Skelett-Koordinatensystems eine Translation in drei Dimensionen möglich: entlang des Richtungsvektors der Kamera, des orthogonalen Vektors zum Richtungsvektor und der Y-Achse der Szene. Die Geschwindigkeit der Translation wird dabei relativ zur Entfernung der linken Hand zu ihrer Ausgangsposition berechnet. Damit nicht jede kleine Abweichung von der Ruheposition zu einer Translation führt, muss ein bestimmter Grenzwert überschritten werden, bevor es zu einer Eingabe kommt. Dieser Grenzwert wird von Eingaben abgezogen, um eine präzise Kontrolle zu ermöglichen. Als Rückmeldung für die Translation dienen eine Geschwindigkeitsanzeige und ein virtueller Analogstick am unteren Rand des Displays. Analog zur Translation erlaubt die Bewegung der rechten Hand entlang der Y- und X-Achse des Skelett-Koordinatensystems die Rotation des Richtungsvektors der Kamera um dessen orthogonalen Vektor und der Y-Achse der Szene. Die Stärke der Rotation ergibt sich ähnlich wie bei der Translation aus der Distanz der rechten Hand zu ihrer Ausgangsposition minus eines Grenzwerts. Die Rotation wird durch einen Pfeil ausgehend vom Zentrum des Displays in Richtung der Rotation visualisiert.
Zur korrekten Verwendung der Lenkungstechnik muss die Z-Achse der Kinect parallel zum Boden ausgerichtet sein. Die noch zu implementierende Lösung wäre die Rotation des Skelett-Koordinatensystems basierend auf einer erkannten Bodenfläche.

3.2 NAVIGATIONSMENÜ

Das Menü besteht aus zwei Einträgen, einen für das Speichern und einen für das Wiederherstellen der Position sowie der Orientierung. Zur Verwendung des Menüs werden die Positionen der Hände auf die Auflösung des XML3D-Canvas projiziert. Befindet sich eine Hand im geöffneten Zustand, wird sie als blauer Punkt mit der zugehörigen Benutzer-ID visualisiert. Zum Öffnen des Menüs müssen beide Hände im geöffneten Zustand zusammengeführt werden. Die projizierte Position der linken Hand wird dabei gespeichert und das Menü relativ dazu auf dem HTML5-Canvas-Element gezeichnet, wodurch die Menüeinträge mit der rechten Hand erreichbar sind (siehe Abbildung 6).
Das Menü wird geschlossen, wenn sich die linke Hand zu weit von der Ausgangsposition entfernt oder wenn beide Hände geschlossen werden. Befindet sich die rechte Hand über einem Eintrag des geöffneten Menüs wird dieser farblich markiert. Die Auswahl kann daraufhin über die Greifgeste erfolgen. Diese basiert auf dem Schließen der Hand, die innerhalb eines bestimmten Zeitraums wieder geöffnet werden muss. Damit der Benutzer weiß, wann die Hand wieder geöffnet werden soll, wird die farbliche Markierung des ausgewählten Eintrags von gelb auf rot geändert. Wartet der Benutzer zu lange mit dem Öffnen der Hand ändert sich die Farbe der Markierung wieder auf Gelb, was signalisiert, dass für eine Selektion die Greifgeste neu begonnen werden muss.

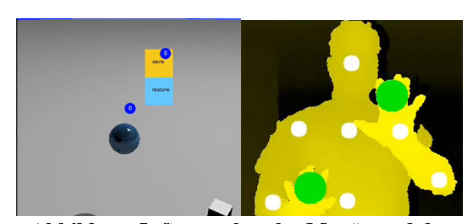


Abbildung 5: Screenshot des Menüs und der korrespondierenden Position des Benutzers mit geöffneten Händen



4 LEISTUNGSTEST

Beide Plug-ins und die in JavaScript implementierte Navigationsanwendung wurden erfolgreich mit einem und zwei Benutzern in RTfox[7] 3.5.4, Chromium 13.0.764 und Firefox 10.10 getestet. Tabelle 1 zeigt die Frameraten im Leerlauf ohne Benutzer in einer leeren Szene und Tabelle 2 die Frameraten während der Navigation mit einem Benutzer ebenfalls in einer leeren Szene. Die Tests wurden auf einem Windows 7 System mit einem 2 x 2,53 GHz Prozessor und 4GB Arbeitsspeicher durchgeführt. Bei dem Microsoft Kinect SDK wurde Version 1.0.0.2 verwendet und bei OpenNI/NITE die Version 1.3.2.3/1.4.1 mit dem PrimeSense Kinect Treiber in der Version 5.0.0.3.4.
Die Dauer der Tests und die durchgeführten Aktionen variieren, weshalb diese Werte untereinander schwer vergleichbar sind. Sie zeigen jedoch, dass eine flüssige Steuerung möglich ist.

5 FAZIT UND AUSBLICK

In dieser Arbeit wurde je ein Webbrowser-Plug-in als Wrapper für die Skelett-Tracker von OpenNI/NITE und der Beta des Microsoft Kinect SDKs erstellt. Die Plug-ins erlauben JavaScript-Anwendungen Daten eines Skelett-Trackers zu nutzen, der um eine Unterscheidung offener und geschlossener Hände sowie der Erkennung einer Greif- und einer Drückgeste erweitert wurde. Zur Demonstration der Laufähigkeit, wurde eine Navigationsanwendung für XML3D-Szenen in JavaScript implementiert, die eine Lenkungstechnik mit fünf Freiheitsgraden und ein leicht erweiterbares Menü zum Sichern und Wiederherstellen des virtuellen Standpunkts bietet. Durch Messungen der Framerate wurde zudem gezeigt, dass eine flüssige Steuerung möglich ist. Mit der Navigationsanwendung wurden bisher nur subjektive Benutzertests durchgeführt. Die korrekte Verwendung der Lenkungstechnik und des Navigationsmenüs war dabei nur mit OpenNI/NITE möglich, da mit der Beta des Microsoft Kinect SDKs zum Entwicklungszeitpunkt Schwankungen der Handkoordinaten auftraten. Bei den Tests der Lenkungstechnik wurde jedoch deutlich, dass sich beide Hände je

[7] Modifizierte Firefox-Version mit einer Raytracer-Engine: http://www.xml3d.org/downloads/

Tabelle 1: Durchschnittliche Framerate und Standardabweichung im Leerlauf ohne Nutzer

	Firefox	Chromium	RTfox
OpenNI	30.0 (0.3)	29.9 (0.4)	29.9 (0.3)
MS Kinect SDK Beta	29.8 (0.9)	29.2 (1.2)	29.7 (0.7)

Tabelle 1: Durchschnittliche Framerate und Standardabweichung während der Navigation

	Firefox	Chromium	RTfox
OpenNI	29.6 (0.7)	29.8 (0.6)	28.3 (1.7)
MS Kinect SDK Beta	27.2 (4.5)	22.7 (2.9)	26.9 (1.8)

nach Position während der Initialisierungspose gegenseitig behindern können. Um neben der Navigation auch Interaktion mit der Szene zu ermöglichen könnte das Menü um kontextsensitive Menüeinträge erweitert werden, z.B. zum Wechseln in einen Manipulationsmodus oder zum Starten von Animationen. Eine Weiterentwicklung des Webbrowser-Plug-ins sollte mit OpenNI/NITE durchgeführt werden, da diese Version eine Übertragung auf Linux-Plattformen erlaubt und somit eine gewisse Plattformunabhängigkeit bietet, was eine der Motivationen von XML3D darstellt.

6 LITERATURVERZEICHNIS

[1] Bowman, Doug A. et al. 2005. 3D User Interfaces: Theory and Practice. Addison-Wesley.

[2] Bradski, G. R. and Kaehler, A. 2008. Learning OpenCV. Computer vision with the OpenCV library. O'Reilly.

[3] Manresa, C. et al. 2000. Real-Time Hand Tracking and Gesture Recognition for Human-Computer Interaction. Electronic Letters on Computer Vision and Image Analysis.

[4] Mozilla Plug-in Dokumentation: https://developer.mozilla.org/En/Plugins. Letzter Zugriff: 10.03.2012

Konvertierung von 3D-Szenendaten am Beispiel von 3ds Max zu Xml3D

Christoph Meißner
Hochschule Reutlingen
Medien und Kommunikationsinformatik
Christoph.Meissner@Student.Reutlingen-University.de

ABSTRAKT

Einhergehend mit der Entwicklung neuer Techniken und neuer Formate ist die Konvertierung ein zeitloses Problem in der Informatik. Insbesondere für virtuelle Umgebungen und diesen zu Grunde liegenden 3D-Szenen stellen sich in diesem Kontext stets neue Herausforderungen, die es zu lösen gilt. Dieser Artikel beschreibt dabei auftretende Probleme und mögliche Herangehensweisen zu deren Lösung. Dabei wird zudem gezeigt, dass Kompromisse erforderlich sind und somit eine verlustfreie Konvertierung nicht in allen Fällen möglich ist.

1 EINLEITUNG

1.1 MOTIVATION

Im Umgang mit 3D-Inhalten – wie Szenen und Modellen – besteht ggf. die Notwendigkeit, diese in unterschiedlichen Anwendungen zu verwenden. Aufgrund spezifischer Formatierungen ist dies jedoch nicht ohne weiteres möglich. In Folge ist eine Konvertierung in ein anderes Format oder die Vereinheitlichung des Formats erforderlich.

Vom DFKI (mit Unterstützung von Intel) wird Xml3D als 3D-Beschreibungsstandard bzw. als einheitliches Format für 3D-Inhalte entwickelt. Das Augenmerk liegt dabei insbesondere auf der Kompatibilität zu verschiedenen Geräten. Mit der Verbreitung verschiedener Plattformen stellt der Browser eine gemeinsame gleichartige Anwendung dar, die es ermöglichen kann, 3D-Inhalte auf beliebigen Geräten darstellen zu können. Der bisherige Fokus lag in der Adaption an verschiedene Renderverfahren, wie WebGL, den hauseigenen Raytracer sowie auch das serverbasierte Rendern, um entsprechend der Leistung des Geräts die Anzeige zu ermöglichen. Dabei liegt das Augenmerk auch auf der einheitlichen Ausgabe der unterschiedlichen Renderverfahren.

Auch die Erstellung von 3D-Inhalten spielt eine Rolle: Xml3D ist zwar textuell vom Menschen lesbar und auch zu bearbeiten, jedoch ist dies nur bis zu einer bestimmten Komplexität möglich, die bei aufwändigeren Szenen schnell erreicht ist. Zudem sind dazu tiefergehende Kenntnisse in der Computergrafik erforderlich. In Folge arbeiten Laien und Grafikdesigner beim Modellieren, beim Umgang mit Texturen, Effekten, Lichtern und Animation mit 3D-Editoren wie Cinema4D, Blender oder auch 3ds Max, die ihre eigenen Formate mit sich bringen. Folglich ist eine Konvertierung vom Ausgangsformat nach Xml3D erforderlich.

1.2 ZIELSETZUNG

Wie zuvor bereits angedeutet, spielt zum einheitlichen Umgang mit 3D-Inhalten die Konvertierung eine wichtige Rolle, wobei deren Art und Weise zu betrachten ist. 3D-Szenen werden zeitpunktabhängig von mehreren Komponenten beschrieben: Darunter fallen Primitive, welche Meshes und somit die dargestellte *Geometrie* beschreiben. Zudem fallen darunter *Shader*, welche mithilfe von Licht, Schatten, Beleuchtungsmodellen und Texturen das Äußere der Geometrie beschreiben [1]. Zur Konvertierung in ein anderes Format sind diese zu übertragen. Eine Konvertierung kann *verlustfrei* oder *verlustbehaftet* auf *exaktem* oder *sinngemäßem* Wege stattfinden. In diesem Kontext ist es ein Ziel, die Problemstellen von 3D-Inhalten zu identifizieren, am Beispiel darzustellen und mögliche Lösungsansätze aufzuzeigen. Dazu werden im Folgenden vorweg auch technische Aspekte betrachtet.



2 TECHNISCHE ASPEKTE

Zur Konvertierung der Szenendaten sind aus technischer Sicht zwei Vorgehensweisen möglich: Zum einen kann die Übersetzung mittels eines bestehenden Konverters in ein einheitliches *Zwischenformat* (wie Xml3D [4] oder VRML [5]), von welchem letztlich in das Zielformat übersetzt wird, stattfinden. Zum anderen kann der Ansatz verfolgt werden, *direkt* in das Zielformat zu übersetzen.

Ein *Zwischenformat* kann zwar in andere Formate übersetzt werden, stellt allerdings einen *Flaschenhals* für von diesem nicht berücksichtige Informationen dar. Somit kann es zu einer verlustbehafteten Konvertierung führen und folglich zu einem fehlerhaften oder unvollständigen Resultat. Zudem verhindert dies die direkte zielformatspezifische Einflussnahme des Nutzers, da etwa zur Ausgangsszene zusätzlich hinzugefügte zielformatspezifische Informationen verloren gehen. Letztlich ist im Fall einer *verlustbehafteten* Konvertierung der *direkte* Weg zu bevorzugen.

2.1 METHODIK

Auch die Übersetzungsmethodik bietet verschiedene Ansatzpunkte je nach Datengrundlage: So kann von einem Format direkt gemäß der Spezifikation des Ausgangsformats (z. B. VRML [5] oder Xml3D [4]) übersetzt werden oder, falls die Spezifikation nicht vorliegt, mithilfe einer Dateninterpretation (z. B. bei 3ds Max-Szenen).

Für den letztgenannten Fall stehen dafür am Beispiel von 3ds Max MaxScript oder MaxSDK zur Verfügung [3], die objektorientierten Zugriff auf den Szenengraphen gewähren. Bei diesen handelt es sich zum einen um eine Skriptsprache (MaxScript) im Rahmen einer szeneninterpretierenden Umgebung und zum anderen um eine native API (MaxSDK), die die Szenendaten interpretiert.

2.2 AUSGABE

Im Sinne der Methodik stellt die Ausgabe in das Zielformat ebenfalls ein Festhalten gemäß einer spezifischen Interpretation im Rahmen der Spezifikation des Zielformats dar. Hierbei ist die Neuinterpretation im Rahmen der Konvertierungsroutine umzusetzen. Hinsichtlich dessen und da Zielformate ggf. aktualisiert werden, ist die Wartung aus softwaretechnischer Sicht ein wichtiger Aspekt.

In Folge bietet es sich an, in der Architektur des Konverters die Ausgabemöglichkeit in ein externes Modul zu verlagern, sodass das Konvertermodul nur die Interpretation vornimmt. Einerseits lassen sich auf diese Art und Weise unabhängig von der gewählten Methodik beliebige Techniken (z. B. Skriptsprachen oder Laufzeitumgebungen wie CLR [2]) einsetzen. Andererseits bietet sich dadurch wieder die Möglichkeit der Vereinheitlichung bezüglich verschiedener Zielformate, indem dieses Modul Szenen zielformatunabhängig abbildet und abstrahiert. Hierfür könnte ein Format wie Xml3D die Grundlage bilden. Dass hierbei kein Informationsverlust auftritt, bedarf jedoch noch weiterer Forschungsarbeit und wird nicht weiter betrachtet.

3 KONVERTIERUNG

Die Konvertierung stellt eine Übersetzung semantisch bedeutsamer Daten dar. Diese finden sich bei 3D-Szenen in einem Szenengraphen (o. Ä.), der die Szene beschreibt, wieder. Ein solcher Graph lässt sich elementweise betrachten, wobei jedes Element ein zu konvertierendes Objekt oder dessen Eigenschaften darstellt, die bei der Ausgabe interpretiert werden müssen. Darunter fallen u. a. die Geometrie, Lichter, Shader und weitere Elemente. Abbildung 1 veranschaulicht dazu einen solchen Graphen anhand zweier Beispielobjekte und deren Translationseigenschaften (darunter

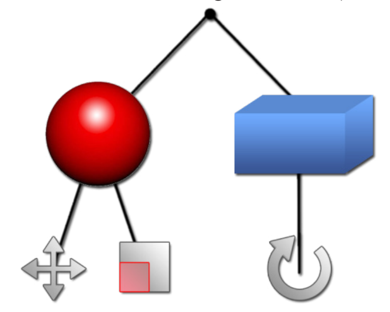


Abbildung 1: Szenengraph

symbolisch dargestellt). 3ds Max bietet dazu z. B. eine objektorientierte Darstellung dieses Graphen [3].

3.1 GEOMETRIE

Eine 3D-Szene wird in einem Koordinatensystem mithilfe von Punkten (Vertices) beschrieben. Dieses Koordinatensystem kann links- oder rechtshändig sein, was bei der Übertragung zu berücksichtigen ist. Die Geometrie der Objekte (Meshes) einer 3D-Szene wird von zusammengehörigen Punkten im Raum (Flächen/Primitive) gebildet. Eine Fläche lässt sich durch zumindest drei Vertices (Triangle) beschreiben. Darüber hinaus können auch vier oder mehr Vertices eine Fläche beschreiben. Auch spielt die Präzision der Vertices hier eine Rolle, wobei einfache und doppelte Präzision möglich sind. [6]

In der Computergrafik werden allerdings in der Regel drei Vertices in einfacher Präzision – aufgrund der damit verbundenen Performanz in Berechnungen – verwendet [1]. Im Regelfall ist somit ein verlustfreies Übertragen von Meshes möglich. Bei Sonderfällen hingegen, wenn die Zahl der Vertices einer Fläche sich im Ausgangs- oder Zielformat unterscheiden, ist eine Aufteilung oder Neubildung der Flächen erforderlich, weshalb hier nur eine verlustbehaftete/sinngemäße Übertragung möglich ist. Im weiteren Sinne ist auch für eine unterschiedliche Präzision nur eine verlustbehaftete/sinngemäße Übertragung möglich: So geht dabei ggf. Genauigkeit verloren, wenn Werte von doppelter in einfache Präzision übertragen werden. Wenn anderenfalls Werte von einfacher in doppelte Präzision übertragen werden, handelt es sich um eine verlustfreie Übertragung.

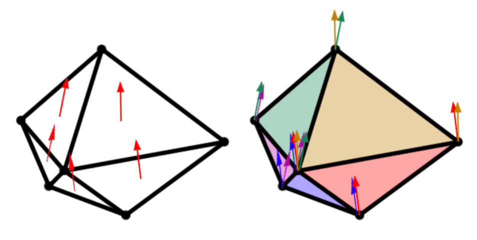


Abbildung 2: Normalen von Triangles

Wie Abbildung 2 veranschaulicht, stellen in der Geometrie die Normalen, welche die Orientierung einer Fläche beschreiben, ein weiteres Problem dar: Sie können dazu auf der Fläche selbst definiert sein (Abb. links – geringste Informationsdichte) oder auch auf den Vertices der jeweiligen Fläche (Abb. rechts), wobei hier je Vertex eine Normale oder je Fläche und Vertex eine Normale (höchste Dichte) definiert sein können. Auch hierbei gilt, dass bei einer Übertragung von Normalen aus geringer in hohe Dichte durch Replizierung eine verlustfreie Übertragung möglich ist, anderenfalls kann höchstens eine sinngemäße Übertragung stattfinden.

3.2 SHADER

3.2.1 TEXTUREN

Texturen können auf unterschiedliche Art und Weise beschrieben und zugeordnet werden. So können Texturen prozedural beschrieben werden oder im Raster (als Bild) vorliegen, wobei eine Konvertierung hin zu einer prozeduralen Beschreibung nur schwer möglich ist. Mittels 2D oder allgemeiner 3D Koordinaten werden sie auf ein Mesh zugeordnet (z. B. Abbildung 3), wobei auch hier eine Konvertierung von 3D zu 2D Koordinaten mit Informationsverlust verbunden ist.

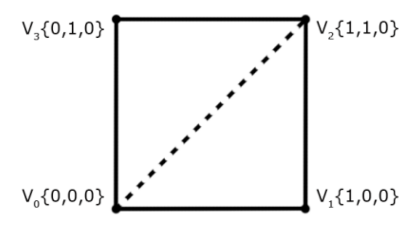


Abbildung 3: Texturkoordinaten

Zudem kann diese Zuordnung parametrisch oder koordinatenweise stattfinden und bei 2D Koordinaten einer spezifischen Rotation (UV, VW, WU) unterliegen. Allgemein gilt wieder, dass eine Übertragung von geringerer Informationsdichte zu einer höheren verlustfrei möglich ist, umgekehrt – von einer höheren zu einer geringeren – jedoch nur verlustbehaftet/sinngemäß übertragen werden kann.

3.2.2 *BELEUCHTUNG UND LICHT*

Die Beleuchtung stellt allgemein eine besondere Herausforderung bei der Konvertierung dar. In der Computergrafik kommen zu deren Berechnung verschiedene Modelle in unterschiedlichsten Kombinationen zum Einsatz, weshalb hierbei eine direkte Übertragung nicht möglich ist. Da diese Berechnungsmodelle oftmals Bestandteil proprietärer Anwendungen (wie 3ds Max) sind und somit nicht exakt bekannt sind, kann zur Übertragung – im Sinne einer heuristischen Problemlösung – an eine solche Kombination nur bestandteilweise approximiert werden. Zudem müssen dabei im Zielformat nicht vorhandene Bestandteile ggf. ergänzt oder ersetzt werden. Folglich ist auch hier nur eine sinngemäße Konvertierung möglich.

Am Beispiel der Konvertierung einer 3ds Max Szene in eine Xml3D Szene bedeutet das, dass 3ds Max beispielsweise zur Abbildung des Glanzpunkts verschiedene Modelle unterstützt. Darunter fallen Phong, Blinn, Blinn-Phong u. a. [3], wohingegen Xml3D derzeit lediglich das Phong-Modell unterstützt [4]. So ist etwa zur Übertragung von Blinn zu Phong nur eine Approximation möglich, wobei jedoch nie dasselbe Ergebnis erreicht wird. Im weiteren Sinne stellt an diesem Beispiel auch die Übertragung von Phong (3ds Max) zu Phong (Xml3D) ein Problem dar, da die Wertebereiche für die Intensität auf jeweils andere nicht-lineare Skalen umgerechnet sind. Abbildung 4 veranschaulicht diesen Zusammenhang.

Ein anderes Beispiel dafür sind auch Lichtquellen, wobei hier die Intensität zum Beispiel in 3ds Max in cd (Candela) gemessen wird [3] und in Xml3D im Prozentanteil zu einer nicht genauer spezifizierten maximalen Intensität [4].

Weitere Unterschiede in den Berechnungsmodellen stellen hierbei auch Lichtfarben sowie das Lichtmodell dar, welches konstant oder abfallend sein kann. So wird in Xml3D der Abfall explizit in der Form $1 / (a_0 + a_1 * d + a_2 * d^2)$ (mit a_x Abschwächungsfaktoren und jeweiliger Distanz d) [4] beschrieben, wohingegen in 3ds Max ein nicht genauer spezifizierter impliziter Abfall der Lichtintensität vorliegt. Auch Schatten, die beispielsweise von einem Raytracer implizit ermittelt werden, sind bei anderen Renderverfahren ggf. ein Problem.

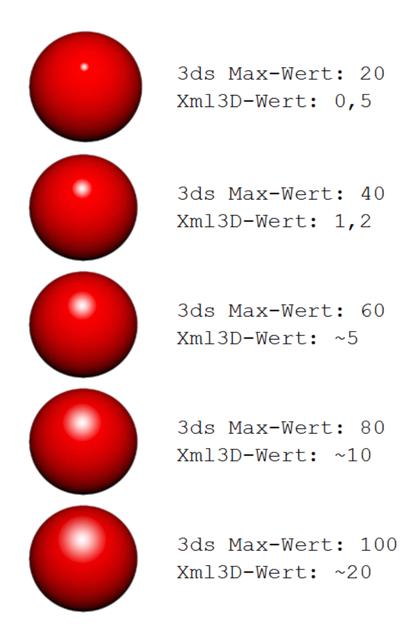


Abbildung 4: Phong-Intensitätswerte

3.3 PROBLEMSTELLEN

Wie zuvor dargestellt, liegen hinsichtlich der formatspezifischen Interpretationen diverse Problemstellen vor, die zu identifizieren sind, indem Ausgangs- und Zielausgaberesultat im Vergleich betrachtet werden. Abgesehen von identischen Resultaten im Rahmen einer *verlustfreien* Konvertierung können hier zwei mögliche Fälle eintreten, die auch komponentenweise zu betrachten sind:

1. Wenngleich 3D-Daten *gleichartig beschrieben* werden können, können deren formatspezifische Interpretationen dennoch zu einem unterschiedlichen Ausgaberesultat führen. Um hier eine korrekte Übersetzung vornehmen zu können, ist folglich eine *sinngemäße* Konvertierung erforderlich.
2. In dem Fall, in dem 3D-Inhalte *nicht gleichartig beschrieben* werden können, kann nur eine *verlustbehaftete* Konvertierung vorgenommen werden, da dabei vorgenommene Umformungen ggf. nicht reversibel sind. Diese lassen sich allenfalls im Rahmen einer *sinngemäßen* Konvertierung annäherungsweise reproduzieren.

Ziel der Konvertierung ist es, dass das Ausgaberesultat beider Renderer subjektiv nicht voneinander zu unterscheiden ist. Abbildung 5 stellt die Ausgaberesultate einer Szene zweier unterschiedlicher Renderer ([a] 3ds Max [b] Xml3D RayTracer) dar. Diese verdeutlichen insbesondere, dass hier die Shader ein Problem darstellen.

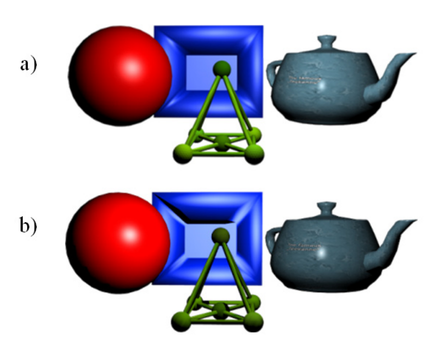


Abbildung 5: Zwei Interpolationen

4 NUTZEREINFLUSS

Zuvor wurden eine Auswahl verschiedener Konzepte und Modelle zur Beschreibung von 3D-Szenen im Kontext zu deren Konvertierung vorgestellt und Problemstellen angesprochen. Ergebnis war, dass hinsichtlich einer singemäßen Konvertierung z. T. nur eine subjektive Annäherung möglich ist.
Dahingehend ist ein weiterer wichtiger Bestandteil der Konvertierung die Möglichkeit für den Nutzer, Einfluss auf diese zu nehmen, wobei zudem zielformatspezifische Optionen berücksichtigt werden können. Darunter fallen auch zusätzliche Informationen, die zum Ausgangsformat hinzugefügt wurden und im Rahmen der Konvertierung in das Zielformat übernommen werden.

5 FAZIT

Es wurde aufgezeigt, dass in der Computergrafik insbesondere verschiedene Konzepte und Modelle eine Überführung von Szenen eines Formats in ein anderes erschweren. Doch auch wenn diese sich gleichen, mangelt es diesen oft insbesondere im Fall proprietärer Implementierungen an objektiven Referenzen. Unabhängig davon ist allenfalls eine subjektive Approximation, die dennoch auch nicht in allen Fällen zielführend ist, möglich. Ebenso wie die Einflussnahme des Nutzers gewährleistet sie keine exakte Lösung dieses Problems, was nicht zuletzt zum Beispiel nichtlinearen Skalen und nicht kompatiblen Modellen geschuldet ist.
Eine kurzfristige Teillösung des Problems im Rahmen der Konvertierung wären objektive Referenzgrößen sich gleichender Modelle, die unabhängig von anderen Modellen durch Vergleichsergebnisse ermittelt werden können und zur Übersetzung dienen. Anderenfalls kann längerfristig nur ein Standard, der benutzerdefinierte Berechnungsmodelle unterstützt, Abhilfe schaffen.

DANKSAGUNG

Besonderer Dank gilt Kristian Sons für Unterstützung und Einblicke hinsichtlich Spezifikation und Funktionsweise der Renderverfahren, insbesondere des Xml3D-Raytracers.

6 LITERATURVERZEICHNIS

[1] Brüderlin Beat, Meier Andreas; Computergrafik und Geometrisches Modellieren; Teubner-Verlag; 2001

[2] http://msdn.microsoft.com/en-us/magazine/cc163567.aspx; CLR Hosting APIs; msdn.microsoft.com; Letzter Zugriff: 20.02.2012

[3] http://www.autodesk.com/3dsmax-maxscript-2012-enu; MaxScript & MaxSDK Dokumentation; www.autodesk.com; Letzter Zugriff: 20.02.2012

[4] http://www.xml3d.org/xml3d/specification/0.4/; Spezifikation von Xml3D (Version 0.4); www.xml3d.org; Letzter Zugriff: 20.02.2012

[5] VRML97, ISO/IEC 14772-1:1997; ISO; 1997

[6] Lengyel Eric; Mathematics for 3D Game Programming & Computer Graphics, Second Edition; Charles River Media; 2004

Pattern Recognition in Gait Analysis with the Ground Reaction Force

Frank Griesinger
Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Frank.Griesinger@Student.Reutlingen-University.de

ABSTRACT

The aim of this study was to test two methods for recognizing gait patterns of people imitating animals. Participants walked on a ground reaction force sensing floor while playing the game Animal Magic, which was developed as prototype during this study. The first method was an approach based on the Hidden Markov Model (HMM). The second method was to use an artificial neural network (ANN) with wavelet transformation for the feature extraction. Results have shown that the patterns were recognized up to 99% with ANN and up to 81% with HMM.

1 INTRODUCTION

The smartfloor is a load-sensing user interface. The main part is the modular floor. Each square module has 4 load sensors, one on each corner. Images are projected onto a screen running along one edge as well as onto the floor.

This novel user interface technology was developed in the Centre for Sports Engineering Research (CSER) of Sheffield Hallam University. For the evaluation of this technology a game called Animal Magic was to be developed.

Animal Magic is intended to encourage preschool children to explore ways of moving. Players are instructed to move like a certain animal. The player's movements are recorded by the meaning of the ground reaction force (GRF) and analyzed and assigned to an animal class. The player sees himself as an animal avatar on the screen. Machine learning mechanisms are used to analyze and classify the player's gait. Those mechanisms are described in this article. Also the results of the tests are shown here.

2 RELATED WORK

Similar projects to the smartfloor are e.g. the load-cell-based Active Floor [4] and video-based approaches like in [3].

Artificial neural networks (ANN) have been previously used for gait analysis [5], recently for example to classify unhealthy gaits. [7] The combination of ANN and wavelet transformation have been explored before, e.g. in speech emotion recognition. [9]

3 THE GAME

The task of the game is to imitate the movement of certain animals, which are defined in 8 classes: bird (1), dog (2), elephant (3), frog (4), horse (5), human (6), mouse (7) and standing (8). The aim of the class "human" was to test if normal movement could also be distinguished from the animal imitations. "Standing" should train the machine learning method to know, when the user stops imitating an animal and just stands still. Though standing should mean a stationary signal, this isn't true due to normal body movement, e.g. breathing, postural sway.

4 PATTERN RECOGNITION

4.1 FEATURE EXTRACTION

The data captured by the smartfloor is the ground reaction force (GRF) that occurs while a person moves on the smartfloor. In this study only the vertical component of the GRF is used.

4.1.1 FEATURE MATRIX

To interpret the signal, its features were extracted by wavelet transformation and a feature matrix. The feature matrix consisted of the mean, the standard deviation and the slope,

of each window the signal was segmented in. This approach was proposed and described by Headon et al. [4] for gait analysis.

4.1.2 WAVELETS

Wavelets are often called a mathematical microscope [6], because they can be used to analyze a signal at different scales. Chau [1] summarizes the main task of Wavelet transformation in signal processing as smoothing and discrimination. However, in this study the aim of using wavelets is to be able to extract the coefficients of a signal that describe its essential character. So it is possible to give the machine learning method only the information that is needed to find the main characteristic, to discriminate different signal patterns. Unused information, e.g. the wavelet coefficients that describe the fine details of a signal, can be ignored.

Whereas in the Fourier transformation the analysis is either in the time or frequency domain the wavelet has a time-frequency representation of the signal. Because of the scaling you know how a frequency was represented at a certain time.

For the wavelet transformation a "mother wavelet" is stretched or compressed to analyze the signal in a certain scale. A big wavelet gives an approximated image of the signal, whereas smaller wavelets describe finer details. [6]

There are various wavelet types to be used as mother wavelet. Discrete wavelet transformation (DWT) is used in this study. By using orthogonal wavelets, perfect reconstruction is possible and fast algorithms are available. The Haar and the Vaidyanathan wavelets are used for comparison. The Haar wavelet is often called the binary wavelet, using only the first coefficients, a blocky representation of the signal is given. The Vaidyanathan wavelet was optimized for speech coding [2] [8]. Both seem capable of maintaining important features with a low scale.

4.1.3 RAW DATA

The raw data were also used as coefficients. By that we wanted to explore the suitability of wavelets as analysis tool. Since many animal patterns look like scaled versions of each other, the length of an analyzed frame was added as another coefficient. Due to the need for having a signal with a dyadic length for the discrete wavelet transformation, the data frame was re-sampled using an interpolating spline fit to a fixed length.

4.1.4 STEP SEGMENTATION

Before the actual feature extraction, the data were pre-processed after being captured. The raw signal was segmented into steps. A step is from one local minimum below bodyweight to the next minimum below bodyweight, after a maximum above the bodyweight. The algorithm that executed this definition, worked well on most classes, but was problematic for the dog class due to its special role as the only animal imitated on all fours.

4.2 MACHINE LEARNING

Two approaches were investigated for machine learning to build a system that recognizes patterns in gait signals.

4.2.1 HIDDEN MARKOV MODEL

A Hidden Markov Model (HMM) describes a Markov process with hidden states. This is an array of states that have certain probabilities of transition. In this model the states can't be observed, but the emission of the transition – the observed features in the feature matrix can be. So the HMM can be trained with the continuous observation of a signal of a class. By the occurrence of an observation for a state transition, it can be calculated how high the possibility is for the sequence of observations to belong to a certain class. pmtk3[1] for Matlab was used for this.

4.2.2 NEURAL NETWORKS

Artificial neural networks (ANN) are a common method for machine learning and therefore often used in artificial intelligence systems. Generally an ANN consists of an input layer, one or more hidden layers and an out-

[1] http://code.google.com/p/pmtk3/, last visited on 20.03.2012.

put layer. A three-layered ANN with Fourier transformation was used for the assessment of gait patterns by Holzreiter et al. [5]. The input layer is the mask for the features of the signal, mathematically a one-dimensional matrix of floating-point numbers. The hidden layer consists of several neurons which are connected by the activation functions, commonly sigmoid functions. A training algorithm adjusts those functions. For a classification problem, the output layer is a one-dimensional matrix of the output classes that represent the probabilities of the input belonging to a certain class, between 0.0 and 1.0.
ANNs are often trained with backpropagation. Two algorithms were compared: the Levenberg-Marquardt backpropagation (LM) is generally suggested, whereas the Bayesian regulation backpropagation (BR) derives from LM and is optimized to generate a network that generalizes well. [8]
For the evaluation of the success of the ANN it is necessary to distinguish between performance and correctness. Performance means the deviation of the output from the target value. For this the mean squared error is used. The lower the value of the performance the better the recognition worked. The correctness is the binary evaluation, which returns true if the output class that is also the target class got the highest value (maximum 1.0).
The termination condition determines when the training algorithm should stop. It is important for the ANN training to stop early to prevent overgeneralization and overspecialization.
The feature matrix is harder to use in an ANN than in HMM, because we have a fixed amount of input features and those are also tested discretely. Whereas for the HMM consecutive features are observed as continuous series. A possible solution would be to put a fixed number of feature matrices together to one input set for the ANN. For the ANN it would be important to start the observation at a certain point, e.g. the beginning of a new step. We excluded the feature matrix approach of the ANN investigation.

5 IMPLEMENTATION

The pattern recognition was developed in Matlab[2] with the Neural Network Toolbox and pmtk3. The game prototype was developed with C# under .NET. The smartfloor was connected via a serial interface by an USB adapter. The recognition interface was loosely coupled with Matlab to quickly apply changes. The game prototype was developed with XNA and it communicates via the UDP protocol with the recognition tool. This protocol was used to integrate the recognition into more powerful game engines in the future.
Fig. 6 shows the modules of the prototype. The symbol is moving on the virtual grass like the user on the smartfloor. It is meant to be like a mirror. (a) The signal preview shows the steps that are tested. (b) The results are visualized as grey fields continuously changing during the test. White means 100% probability and black means 0% probability. (c) The smartfloor can be controlled with the control window. (d)

6 RESULTS

Data of 22 participants was captured. The participants were randomly split into training and testing sets. A common ratio is 70%/30% (train/test), which we finally used. [8] Holzreiter et al. tested it up to 80%/20% with the best results, with just a marginal improvement of the maximum value, but higher deviations, so many trained ANNs were probably not enough generalized. [5]

Table 1: Comparison of HMM and ANN

	maximum	average
HMM	81.48%	51.99% (61.34%)
ANN (wavelet)	99.21%	98.35%
ANN (raw)	99.05%	98.15%

[2] http://www.mathworks.de/, last visited on 20.03.2012

The results show that the ANN approach worked better with a maximum correctness of 99.21% using the wavelet transformation (Table 1). By using the HMM a system was achieved with a best correctness of 81.48%. The over-all-average was 51.99%. Due to the random initialization (the initial guess for state transitions) there were relevant downward deviations, so this is an approximated result. Without the significant outliers the average is 64.34%. For the results of the ANN shown in Table 1 only the ANN with the most suitable configurations were used.

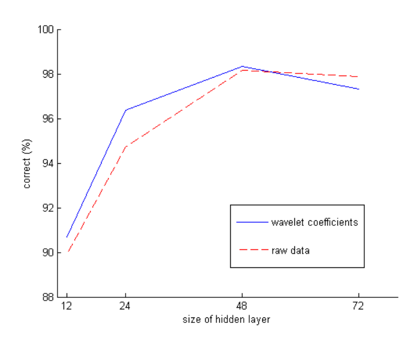
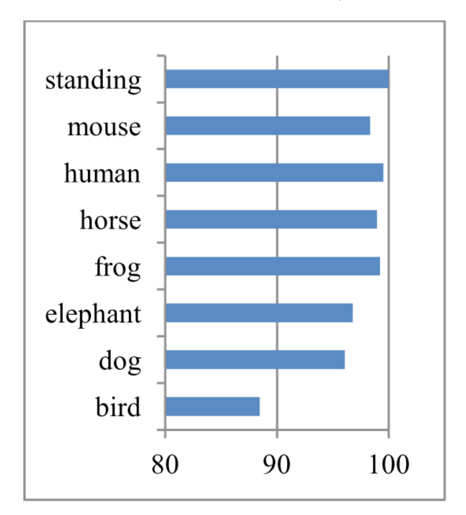


Figure 1: Comparison of different hidden layer sizes

Fig. 1 shows the results of ANNs that use 64 input coefficients with a hidden layer (HL) size of 12, 24, 48 and 72, for a re-sampled signal with a dyadic length of 64. The wavelet coefficients provide generally a better result, especially when the size of the HL is low, whereas the ANN gets better trained with raw data when more neurons are in the HL. Possibly the wavelet transformation highlights the essential features of a gait, which could make it easier for the ANN to recognize it with fewer neurons. However, it was not possible to identify a significant difference between raw data and wavelet coefficients as input values, because both approaches achieved very good recognition results.
A HL size of 72 gave more varying values, but always worse compared to 48 neurons. We assume that the network became overspecialized. No further investigation was done with bigger HL sizes. Table 3 shows that the best configuration for the ANN to analyze 2 steps with a length of 64 could be estimated as 32 input coefficients (31 wavelet coefficients plus the original length of the raw signal l_r), 48 HL size, 800 epochs (of testing) and using the Bayesian regulation backpropagation (*trainbr*, ANN training function).
For a good reconstruction with the important features of the signal, a suitable amount of wavelet coefficients *c* could be computed as $c = (s/2)-1$, where *s* is the dyadic length of the tested steps. So the first *c* wavelet coefficients are used for a satisfactory ANN.
Best results were achieved by testing at least two steps at a time. Also 3 steps gave a suitable result but without any improvements. The advantage in using 2 steps instead of 3 or more is to identify a pattern earlier and not to confuse the ANN when a person changes the imitated animal.
An interesting result of this gait analysis is that the patterns of different people look very similar. Differences evolved in the patterns by varying bodyweights, could be reduced by using normalization.
The tests weren't instructed at first, but the test persons had a very similar understanding of the way how to imitate animal movement, so we decided to give a reference for the imitation. Anyhow there was still some variety in the recorded data, e.g. by direction and intensity of movement. For the smartfloor a frame rate of 25 frames per second (fps) were used. For comparable gait analysis devices often higher rates are used, but for the rough pattern of a gait 25fps seemed reasonable.
The saturation curve to the final value is very steep. But marginal improvements of the performance occur even after thousands of repetitions (epochs). A good value was between 800 and 1000. Fig. 2 shows that a seemingly good correctness is already reached after 200 epochs, though the performance is still improving. The gradient of the performance is therefore a better termination condition than the correctness. In field tests an ANN with a better performance but the same correctness as another ANN with less performance seemed to work better in general.
Holzreiter et al. propose 200,000 repetitions for an ANN with Fourier transformation, but this is depending on the training function and the

Table 2: Correctness of an ANN by the classes



amount of features and neurons in the hidden layer. [5] Because of the good correctness of the ANN, the confusion matrix isn't very surprising (Table 4). But it shows that the imitations of a bird and a dog are the hardest to determine. A bird was more often interpreted as an elephant or a mouse than one of the other wrong classes. However, the results are not general, since those are just a few misidentifications, and the amount of steps to test differs, e.g. for a mouse the imitator does more steps in the same time than for an elephant.

Table 2 shows how well the ANN worked for each class. Standing was always detected, since there is, unlike the other classes, no clear pattern. Obviously it's not good for step segmentation, because there are no steps, however due to postural sway the algorithm determined some steps.

The comparison of the ANN training methods has shown that BR is better suited for the classification than LM. With LM and Vaidyanathan wavelet we achieved up to 69% correctness (average: 42%) and with LM and Haar wavelet up to 49% (average: 31%). Despite the random initialization of the activation functions of the ANN, the training algorithm (BR) worked so well, that we have just a small standard deviation. 5 repetitions showed nearly the same result as 10.

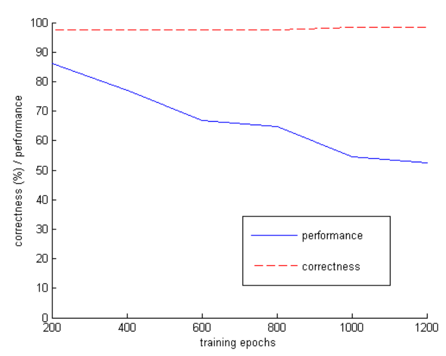


Figure 2: Relation of correctness and performance

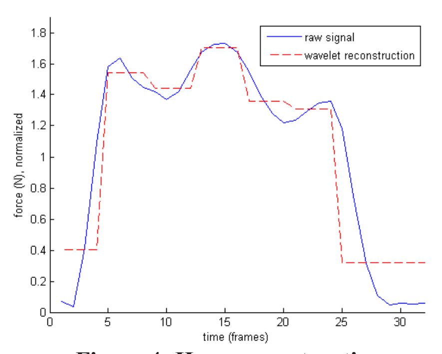


Figure 4: Haar reconstruction

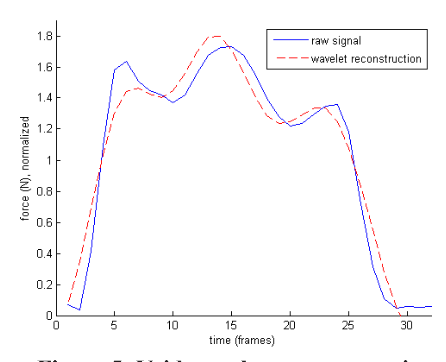


Figure 5: Vaidyanathan reconstruction

Fig. 3 and Fig. 4 show the reconstruction of a signal by using the first 7 wavelet coefficients of a signal with 32 values. The Vaidyanathan wavelet preserves the features better than Haar. In 500 tests an improvement of the correctness by 10% could be achieved. For a general result more tests

with different configuration and more re-initializations should be made. Since the results were sufficient for this task, no further investigations were done.
The results presented here were obtained by tests with recorded data, however people behave differently on field tests. Irregularities make the correctness worse, especially when people change animal class. For this problem the choice for a certain class should not be made after only a few iterations and when more than one class got a probability of over 60%.
As is seen on Fig. 5 the interpretation of imitating a horse is very similar (upper images). The wavelet coefficients don't look easier to recognize a pattern to humans, but the test shows that significant features were encoded by the coefficients. It is also clear to see that the coefficients for the higher scales are insignificant. This is also due to the low sampling rate of 25fps, so little details of a gait are not captured.

7 CONCLUSION

It was shown that the neural network method is very suitable for this task. Wavelet analysis helped the neural network do the classification. However, due to the almost equally good results with raw data the actual enhancement by using wavelets could not be determined entirely. More animal classes should be created and evaluated with a bigger amount of test persons to fully utilize the ANN approach.
The full capability of the smartfloor was not used, because the position of the user was not used in the analysis. By additionally using the provided information of speed and direction of the user's walk, an even better classification may have been achieved.
Research on this topic could also be done in another direction. Here we tried to find raw patterns especially with wavelet analysis. But we could also filter the raw patterns and use those wavelet coefficients that describe the fine characteristics of a gait, e.g. to analyze the gait of post-stroke patients [7].
Currently games are being developed for the smartfloor with a framework based on the game engine Unity. Animal Magic could then get a nice interface and encourage children to move.

ACKNOWLEDGEMENTS

Special thanks to my supervisor Ben Heller for all his help and support. I worked together with Ewan Davies, a student at Jesus College of the University of Cambridge, his main part was the HMM approach, so those are the results of his implementation. I want to thank him as well for his support and especially mathematical help.

8 REFERENCES

[1] Chau, Tom: A review of analytical techniques for gait data. Part 2: neural network and wavelet methods, Gait and Posture 13, p. 102–120, 2001.

[2] Gonon, Gilles et al.: Extended Wavelet Packet Decomposition and Best Basis Search Algorithm, Evaluation of entropic gain for audio signals, Université du Maine, 2001.

[3] Gronbaek, Kaj et al.: iGameFloor, Proceedings of the International Conference on Advances in Computer Entertainment Technology (ACE), 2007.

[4] Headon, Robert et al.: Recognizing Movements from the Ground Reaction Force, Proceedings of the 2001 Workshop on Perceptive User Interfaces (PUI), 2001.

[5] Holzreiter, Stefan et al.: Assessment of gait patterns using neural networks, J. Biomechanics Vol. 26, No. 6, pp. 645-651, 1993.

[6] Hubbard, B. B.: The World According to Wavelets, 1st edition, A K Peters, Massachusetts, 1996.

[7] Kaczmarczyk et al.: Gait classification in post-stroke patients using artificial neural networks, Gait & Posture 30, p. 207-210, 2009.

[8] Matlab Documentation for the Neural Network Toolbox, 2011.

[9] Shah, Firoz: Discrete Wavelet Transforms and Artificial Neural Networks for Speech Emotion Recognition, International Journal of Computer Theory and Engineering (IJCTE), Vol. 2, No. 3, June, p. 319 - 322, 2010.



		Hidden layer size									
		12		24		48		72		96	
		w	r	w	r	w	r	w	r	w	r
Coefficients (without 1_r)	15	0.881 (0.840)	0.883 (0.851)	0.930 (0.913)	0.930 (0.909)						
	31	0.908 (0.863)	0.930 (0.905)	0.979 (0.955)	0.970 (0.953)	0.994 (0.978)	0.992 (0.977)				
	63	0.935 (0.907)	0.924 (0.899)	0.985 (0.964)	0.968 (0.947)	0.992 (0.984)	0.990 (0.982)	0.987 (0.979)	0.992 (0.979)	0.794 (0.773)	0.968 (0.949)
Legend: w = wavelet coefficients; r = raw data. Value: maximum (average).											

Table 3: Comparison of different ANN configurations

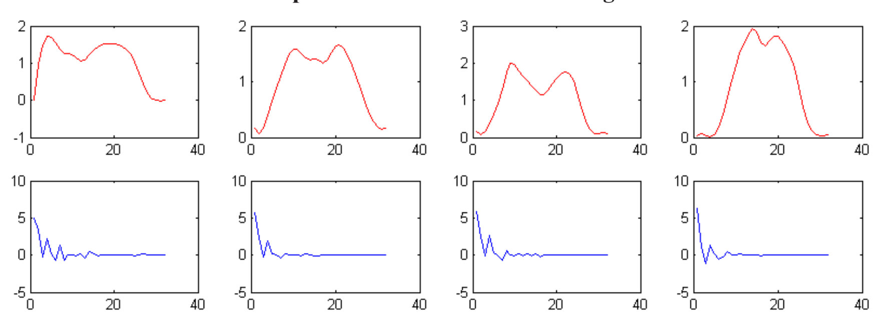


Figure 5: Comparisons of steps of different persons imitating a horse

		target class							
		bird	**dog**	**elephant**	**frog**	**horse**	**human**	**mouse**	**standing**
output class	**bird**	**166**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	dog	**3**	**288**	**4**	**1**	**1**	**2**	**2**	**0**
	elephant	**4**	**1**	**293**	**0**	**0**	**0**	**2**	**0**
	frog	**1**	**0**	**0**	**168**	**1**	**0**	**2**	**0**
	horse	**2**	**0**	**0**	**0**	**348**	**0**	**0**	**0**
	human	**2**	**0**	**3**	**0**	**0**	**255**	**3**	**0**
	mouse	**5**	**6**	**1**	**1**	**1**	**0**	**653**	**0**
	standing	**0**	**0**	**2**	**0**	**0**	**0**	**0**	**74**

Table 4: Confusion matrix of the ANN approach

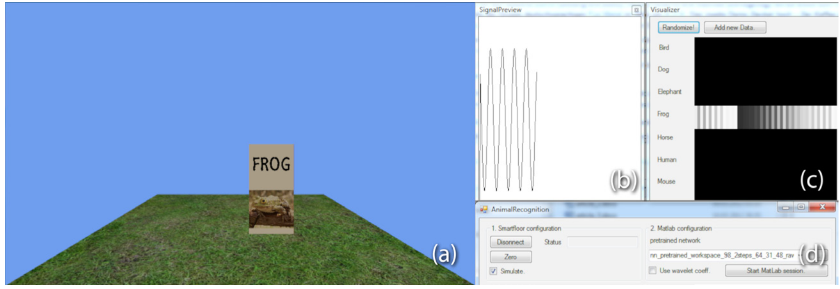


Figure 6: Overview of the prototype: front screen (a), signal preview (b), probability visualizer (c), control window (d).

Interpolation bei Showlaser Systemen mit dem Tracking System Kinect

Michael Schempp
Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Michael.Schempp@Student.Reutlingen-University.de

ABSTRACT

Diese Arbeit betrachtet Ansätze zur Interpolation zwischen zwei Punkten, um mit Hilfe des Tracking-Systems „Kinect“ eine Laserprojektion zu erzeugen. Ein bestehendes und gängiges Verfahren zur Interpolation bei Showlaser-Systemen und ein eigenes Verfahren werden miteinander verglichen und auf Basis einer prototypischen Implementierung bewertet.

1 EINLEITUNG

Grundgedanke dieser Arbeit war mit Hilfe des Tracking-Systems Kinect eine Möglichkeit zu schaffen, Personen zu tracken und mit Showlaser Projektoren Bewegungen der Personen in Echtzeit zu projizieren.

Showlaser-Systeme zeichnen sich dadurch aus, dass sie im sichtbaren Wellenlängenbereich von 400nm (Nanometer) bis 700nm emittieren und hauptsächlich zu Unterhaltungszwecken eingesetzt werden (vgl. [2] S.33ff). Hierbei werden Laserstrahlen mittels schneller Ablenkeinheiten durch den Raum bewegt, um die Trägheit des Auges auszunutzen, damit Laserprojektionen erzeugt werden können (vgl. [1] S.27ff).

Damit sich Ablenkeinheiten nicht zu schnell zwischen einzelnen Punkten bewegen müssen und somit Endpunkte bei zu hoher Geschwindigkeit überfahren, wird zwischen diesen Punkten interpoliert (vgl. [1] S.27). Bei einer einfachen linearen Interpolation bewegen sich die Ablenkeinheiten jedoch immer in der gleichen Distanz zwischen einzelnen Punkten, unabhängig davon, ob es sich um Anfangs- bzw. Endpunkte handelt oder um interpolierte Punkte. Dies bedeutet, dass der Scanner einen Punkt immer mit derselben Geschwindigkeit anfährt, was zu Problemen innerhalb der Projektion, insbesondere bei schnellen Richtungswechseln, führen kann (Siehe 3.1). Um solchen Problemen vorzubeugen, soll eine andere Art der Interpolation zwischen zwei gegebenen Punkten gefunden werden, um Problemen vorhandener Interpolationsalgorithmen entgegenzuwirken.

2 GRUNDLAGEN

In den folgenden Absätzen wird nur zum Teil auf die komplexe Technik und Funktion eines Showlaser-Systems eingegangen . Es wird auf technische Grundlagen eingegangen, die das Thema dieser Arbeit (Laserausgabe und Interpolation) direkt betreffen. Da auf dem Gebiet der Showlaser nur sehr wenig allgemeine Literatur existiert, (vgl. [2]) basieren die folgenden technischen Grundlagen zu Laserprojektionen im Unterhaltungsbereich auf eigenen Erfahrungen und Analysen der zur Verfügung gestellten technischen Anlagen, Software, Handbüchern und Veröffentlichungen der Firma Medialas Electronics, sowie auf einigen Veröffentlichungen und Industrie-Standards der ILDA (International Laser Display Association) und den Büchern von Dirk R. Baur ([1] & [2]).

2.1 LASERPROJEKTIONEN

Die folgenden Informationen basieren auf der Funktion von Laserprojektionen und Software der Firma Medialas (Bsp: [5], [4], [6]) sowie auf dem “ILDA File Interchange Format” [9]. Bei einer Laserprojektion wird ein Bild oder eine Animation mit Hilfe von Laserprojektoren auf eine Projektionsfläche gebracht. Eine Laserprojektion kann aus einem oder mehreren Frames bestehen, welche wiederum aus mehreren, bis zu einigen Tausend, einzelnen

Punkten innerhalb eines 2D- oder 3D-Koordinatensystems besteht. Diese einzelnen Punkte enthalten mehrere Informationen welche für den Laserprojektor wichtig sind z.B.: Farbinformationen, Intensität und X- und Y- Position des Punktes. Es besteht eine Hierarchie innerhalb eines einzelnen Laserbildes,die definiert in welcher Reihenfolge das Ablenksystem des Laserprojektors einzelne Punkte innerhalb eines Frames anfährt. Durch das Zusammenspiel von Bewegung des Ablenksystems und dem An- bzw. Abschalten der Laserquellen innerhalb des Laserprojektors, können somit Figuren entstehen. Diesen Vorgang nennt man „Scannen“. Dies bedeutet, dass die Laserpunkte innerhalb eines Frames entsprechend der Hierarchie mit einer bestimmten Geschwindigkeit (Scangeschwindigkeit wird in pps (points per second) angegeben) angefahren werden und je nach dem, welche Informationen der Punkt bereitstellt, die Laserquellen eingeschaltet, ausgeschaltet, oder gedimmt werden (vgl. [1] S.35ff) (Diesen Vorgang nennt man „modulieren der Laserquelle“).

2.2 KINECT

Das offizielle „Kinect for Windows SDK“ der Firma Microsoft gibt die Möglichkeit bestimmte Gelenkpositionen der getrackten Person direkt abzurufen und somit ein digitales Skelett zu erstellen [11].
Diese getrackten Punkte bieten die Grundlage für die Laserprojektion. Um jedoch ein Abbild des Skeletts zu bekommen, müssen nicht nur die Gelenke, sondern auch deren Verbindungen mittels des Laserstrahls auf die Projektionsfläche gebracht werden.

3 INTERPOLATION BEI VORHANDENER SHOWLASER SOFTWARE

Damit Ablenkeinheiten innerhalb der Projektoren nicht durch schnelles Beschleunigen und Abbremsen zwischen einzelnen Punkten unnötig strapaziert werden, müssen zwischen den Punkten Stützpunkte durch Interpolation eingefügt werden [1].

Die Recherche ergab, dass die Lasershow-Software „Mamba” der Firma Medialas, eine lineare Interpolation implementiert. Auch bei anderer Software ergaben Versuche mit dem Oszilloskop, dass diese ebenfalls lineare Interpolationen implementieren. Bei den Versuchen wurden mit Hilfe der entsprechenden Software zwei Punkte innerhalb der möglichen Projektionsfläche erstellt, über die zugehörigen Hardware-Interfaces ausgegeben und mit Hilfe eines Oszilloskops analysiert (siehe Abbildung 1).
Diese lineare Interpolation fügt Zwischenpunkte mit variablem Abstand zwischen zwei gegebenen Punkten ein. Weiterführende Versuche mit der Software „Mamba” der Firma Medialas ergaben, dass diese Art der Interpolation zu Problemen führen kann, wenn die Software keine entsprechenden Gegenmaßnahmen bereitstellt.

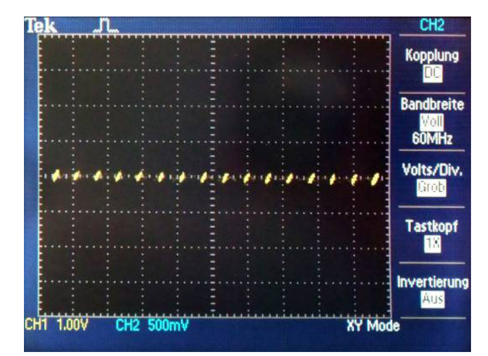


Abbildung 1: Interpolation zwischen zwei Punkten bei der Software „Mamba“

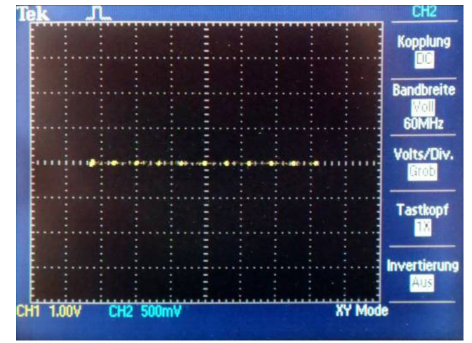


Abbildung 2: Interpolation zwischen zwei Punkten bei der Software „Pangolin“

3.1 ANALYSE DER VORHANDENEN INTERPOLATIONSART

Um den aktuellen Interpolationsalgorithmus zu analysieren, wurde ein weiterer Versuch durchgeführt, bei dem ein Testbild erstellt und mit einem Laserprojektor projiziert wurde. Es wurde kontinuierlich die Scangeschwindigkeit des Projektors um 5.000 pps von 10.000 pps auf maximal 35.000 pps erhöht. Dabei wurden folgende Faktoren überwacht:
Optische Eindrücke, ob die Projektion entsprechend dem Testbild aussah (bewertet durch eine Skala von ----, sehr schlecht bis ++++, sehr gut), die Temperatur des Ablenksystems und die Lautstärke des Ablenksystems. Gemessen wurde bei einem normalen RGB-Laserprojektor inkl. Betriebsgeräusche (Bsp: Lüftung). Um ein aussagekräftiges Ergebnis zu erhalten, wurde bereits 15 Minuten vor der ersten Messung die Projektion mit einer Scangeschwindigkeit von 10.000 pps projiziert, um das Ablenksystem auf die entsprechende Betriebs-Temperatur zu bekommen. Tabelle 1 zeigt die Versuchsergebnisse.

Tabelle 1: Versuchsergebnisse „Mamba“

pps	Temp.	Dezibel	Projektion
10 000	22,8°C	50,6	+
15 000	23,1°C	51,1	0
20 000	23,5°C	51,8	-
25 000	24°C	52,8	-
30 000	25,5°C	53,3	--
35 000	26,9°C	55,2	---

Bei einem Erhöhen der Scangeschwindigkeit des Ablenksystems fahren Scanner schneller X,Y-Positionen an als die Laserquellen moduliert werden. Dies bedeutet, dass die Laserquelle ausgeschaltet wird, während das Ablenksystem bereits auf dem Weg zum nächsten Punkt ist oder den entsprechenden Endpunkt noch nicht erreicht hat. Dies resultierte in Ausreißern bei der Projektion (siehe Abbildung 3). Um diesem Problem entgegenzuwirken, bietet die Software Korrekturen an, damit beispielsweise die Farbinformation der Punkte innerhalb der Projektion um einen variablen Faktor nach vorne oder hinten verschoben werden, um die Laserquelle früher oder später an- bzw abzuschalten, obwohl das Ablenksystem den Endpunkt noch nicht oder schon länger erreicht hat. Da Laserquellen bei verschiedenen Spannungen unterschiedlich schnell ein- bzw. ausschalten (vgl. [10]), kann mit einer globalen Korrekturfunktion für alle Farben dieses Problem nur zum Teil kompensiert werden (gut zu Erkennen bei Abbildung 5).
Durch Erhöhen der Scangeschwindigkeit erhöhte sich auch die Temperatur des Ablenksystems (Siehe Tabelle 1). Durch Erhöhung der Temperatur steigt auch die Gefahr der Zerstörung des Ablenksystems (vgl. [10]). Bei komplexen Frames mit vielen Einzelpunkten, ist eine schnelle Scangeschwindigkeit jedoch wünschenswert, um ein “Flackern” der Projektion zu verhindern [vgl. [1] S.27ff).
Das Geräusch des Ablenksystems erhöhte sich nur minimal, wobei nicht ausgeschlossen werden kann, dass an der Erhöhung der Betriebslautstärke auch andere Bauteile wie z.B. die Belüftung Schuld hat. Vielmehr scheint eine Veränderung der Tonfrequenz, welche die Scanner von sich geben, stattzufinden. Dies kann hier allerdings nur subjektiv beurteilt werden, da eine weiterführende Analyse der Tonfrequenz nicht stattgefunden hat.

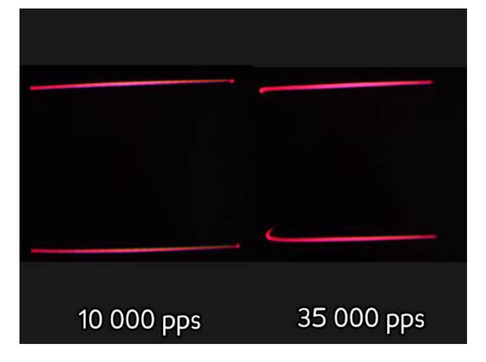


Abbildung 3: Testbild und Vergleich zwischen unterschiedlichen Scangeschwindigkeiten

4 IMPLEMENTIERUNG

4.1 ANFORDERUNGEN

Durch die Analyse der Versuchsergebnisse zur Software „Mamba“ und dem eigentlichen Ziel bzw. Grundgedanken welcher dieser Arbeit zu Grunde liegt, ergaben sich folgende Anforderungen an die prototypische Implementierung:

- Tracking einer Person durch das Tracking-System Kinect.
- Ausgabe eines einfachen Skeletts durch einen Laserprojektor bei Verwendung von Hardware der Firma Medialas.
- Anderer Interpolationsansatz um Zwischenpunkte innerhalb den getrackten Punkten einzufügen, damit problematische Verhaltensweisen, die beim Erhöhen der Scangeschwindigkeit des Ablenksystems mit dem analysierten Interpolationsmechanismen auftreten, vermieden oder verbessert werden können.

4.2 NEUER INTERPOLATIONSANSATZ

Bei dem neuen Interpolationsansatz werden Zwischenpunkte anders als bei der Standard Interpolation der Software „Mamba“, nicht mit einem gleichbleibenden Abstand zwischen den gegebenen Punkten eingefügt, sondern abhängig von der Position des einzufügenden Punktes mit variablem Abstand. Dies bedeutet vor allem, dass Punkte, welche näher an Start- bzw. Endpunkt eingefügt werden müssen, mit einem kleineren Abstand zueinander eingefügt werden als Punkte, welche sich beispielsweise in der Mitte der gegebenen Punkte befinden (Siehe Abbildung 4). Hierzu wird die Strecke zwischen Start- bzw. Endpunkt in verschiedene Zonen eingeteilt, in denen mit unterschiedlichen Interpolationsdistanzen gearbeitet wird. Innerhalb der ersten 10% der Strecke wird mit 1/6 der eingestellten Interpolationsdistanz interpoliert. Zwischen 10% und 20% mit 1/4 und zwischen 20% und 30% mit 1/2. Dies bedeutet, dass zwischen 30% und 70% mit der Standardinterpolation interpoliert wird, zwischen 70% und 80% wiederrum mit 1/2, zwischen 80% und 90% mit 1/4 und zwischen 90% und 100% wiederrum mit 1/6 der Interpolationsdistanz.

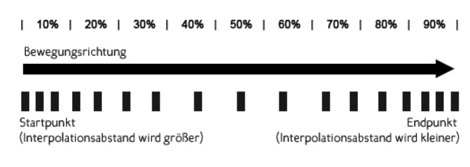


Abbildung 3: Neuer Interpolationsansatz

In der Theorie müsste die Ablenkeinheit somit an kritischen Stellen der Projektion (An Start- bzw. Endpunkt), an denen meistens schlagartige Richtungswechsel stattfinden, durch den immer dichter werdenden Punktabstand langsam abgebremst bzw. beschleunigt werden.
Dies soll zur Folge haben, dass höhere Scangeschwindigkeiten gefahren werden können, die Projektion allerdings keine bzw. weniger Ausreißer bekommt, die Temperatur nicht so stark steigt und sich das Betriebsgeräusch der Ablenkeinheit verändert.

4.3 LASERAUSGABE MITTELS KINECT

Durch das „Kinect for Windows SDK“ ist es möglich, auf bestimmte Trackingpunkte eines menschlichen Körpers (wie z.B. Hände, Füße und Schultern) direkt zuzugreifen (vgl. [12]). Durch das „Coding4Fun Kinect Toolkit“ ist es zudem recht einfach, die getrackten Datenwerte (X- und Y-Position der Trackingpunkte) so zu skalieren, damit sie für die eigene Anwendung brauchbar sind (vgl. [11]). In diesem Fall wird eine X-,Y-Auflösung von 4096 auf 4096 Bildpunkte benötigt, um die volle Ausgabegröße des „Medialas Hyperport Pro DMX“ USB-Interfaces ansteuern zu können. Die Ablenkeinheit besitzt somit eine Auflösung von 4096 Punkten in X- bzw. Y-Richtung, bei einer maximalen Auslenkung von 40° (gemessen an 3 verschiedenen Laserprojektoren der Firma Medialas).
Jeder getrackte Punkt liefert eine X-/Y-Koordinate zurück und bietet somit die Grundlage für die Projektion.

Die Laserausgabe wird über das von der Firma Medialas zur Verfügung gestellte USB-Interface „Hyperport Pro DMX“ realisiert. Hierfür bietet Medialas eine Dynamic Link Libary (DLL) an, welche die entsprechenden Funktionalitäten zur Laserausgabe bereitstellt (vgl. [6]).

Die DLL benötigt zur Ausgabe einen einzelnen Frame, welcher aus mehreren Punkten besteht. Für die Ausgabe der getrackten Punkte werden diese zunächst dem Interpolationsalgorithmus in einer bestimmten Reihenfolge übergeben, um zwischen Start-und Endpunkten interpolieren zu können. Zusätzlich zu den X- bzw. Y-Koordinaten erhalten die Punkte Farb- & Intensitätswerte, um festzulegen in welcher Farbe die Punkte gezeichnet werden sollen bzw. ob diese überhaupt sichtbar vom Laser dargestellt werden sollen. So wird Punkt für Punkt ein Frame zusammengesetzt und anschließend an die Hardware übergeben, welche den Frame an den Laserprojektor weitergibt.

Die Trackingdaten werden nach jedem Fertigstellen eines Frames aktualisiert und erneut in einem aktualisierten Frame festgehalten, welcher wiederum an die Hardware übergeben wird. Dieser Vorgang wird wiederholt, solange das Tracking-System Daten liefert.

5 EVALUIERUNG

Um Aussagen über den prototypisch implementierten Interpolationsalgorithmus treffen zu können, wurde der bereits durchgeführte Versuch zur Analyse der Ausgabe der Software Mamba wiederholt. Um ein möglichst unverfälschtes Ergebnis zu erhalten, wurde das Live-Tracking deaktiviert und ein zuvor aufgezeichnetes Skelett ausgegeben. Zusätzlich, zum neu entwickelten Interpolationsalgorithmus, wurde ein normaler Interpolationsalgorithmus implementiert, wie man ihn auch in der Software “Mamba” findet.

Zunächst wurde der Versuch am normalen Interpolationsalgorithmus durchgeführt und wiederum optische Eindrücke, Temperatur und Lautstärke gemessen, während die Scangeschwindigkeit in 5.000er Schritten von 10.000 pps auf 35.000 pps erhöht wurde.

Anschließend wurde derselbe Versuch am neuen Interpolationsalgorithmus durchgeführt und die Werte aus beiden Versuchen miteinander verglichen. Das Ergebnis ist in den Tabellen 2 und 3 zu sehen.

Tabelle 2: Versuchsergebnisse „Skelett“ Standardinterpolation

pps	Temp.	Dezibel	Projektion
10 000	23,4°C	54,2	0
15 000	23,7°C	54,4	0
20 000	24,6°C	54,6	-
25 000	25,8°C	55,4	--
30 000	27,9°C	56,5	---
35 000	29,3°C	58,5	----

Tabelle 3: Versuchsergebnisse „Skelett“ neue Interpolation

pps	Temp.	Dezibel	Projektion
10 000	22,3°C	52,7	-
15 000	22,4°C	52,8	0
20 000	22,3°C	52,9	++
25 000	22,8°C	52,9	++++
30 000	22,8°C	53,4	++++
35 000	22,9°C	53,6	+++

5.1 OPTISCHE EINDRÜCKE

Während bei der Figur bei normaler linearer Interpolation (ohne Korrekturen) bereits bei 10.000 pps leichte Ausreißer in der Projektion wahrnehmbar waren, gab es beim neuen Interpolationsansatz erst ab 30.000 pps sehr leichte Ausreißer . Die Projektion war jedoch selbst bei 35.000 pps noch sehr gut zu erkennen, während bei der normalen Interpolation die Figur völlig deformiert war (Siehe Abbildung 5). Ecken und Kanten innerhalb der Projektion waren beim normalen Interpolationsansatz bei 35.000 pps nicht mehr wahrnehmbar, beim neuen Interpolationsansatz hingegen schon(Siehe Abbildung 6). Zu erwähnen ist



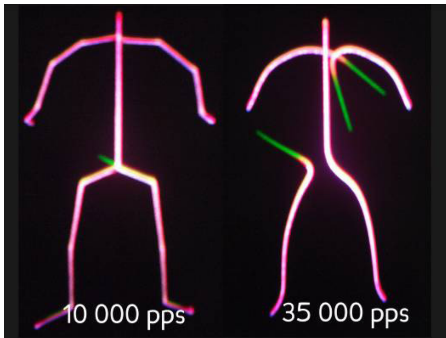


Abbildung 5: Skelett bei verschiedenen Geschwindigkeiten mit Standardinterpolation

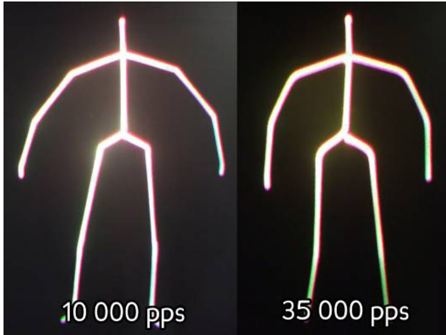


Abbildung 6: Skelett bei verschiedenen Geschwindigkeiten mit neuer Interpolation

jedoch, dass die Projektion bei dem neuen Interpolationsansatz bis 20.000 pps nicht flackerfrei darzustellen war, bei dem normalen Ansatz war hingegen bereits ab 10.000 pps kein Flackern festzustellen. Dies spielt aber aufgrund der Anforderung, hohe Scangeschwindigkeiten zu fahren, keine Rolle.
Aufgrund dessen, dass das Ablenksystem um die Bereiche der Start- bzw. Endpunkte langsam gebremst bzw. beschleunigt wird, konnten hellere Stellen in der Projektion festgestellt werden, da diese Fläche einer längeren Einwirkdauer ausgesetzt war als andere. Dies kann beim Betrachter als störend empfunden werden, ist jedoch nicht das Hauptproblem des Phänomens. Ausschlaggebend für die Unterhaltungs-Branche ist in dieser Hinsicht das Thema Lasersicherheit(vgl. [7] & [8]). Da die helleren Stellen innerhalb der Projektion auch eine höhere Laserleistung besitzen als die dunkleren Stellen (gemessen mit Coherent Fieldmate-Messgerät), wird es schwieriger etwaige MZB-Werte innerhalb von Veranstaltungsorten einzuhalten und somit schwieriger eine Einzelabnahme von Sachverständigen für solche Installationen zu erwirken.

5.2 TEMPERATUR

Bei dem neuen Interpolationsansatz konnte kein nennenswerter Temperaturanstieg festgestellt werden (Siehe Tabelle 3). Bei normaler Interpolation hingegen war ein Temperaturanstieg von fast 9° feststellbar, was die Vermutung bestätigt, dass der neue Interpolationsansatz das Ablenksystem nicht so sehr strapaziert, wie der andere Interpolationsansatz.

5.3 LAUTSTÄRKE

Anders, als in den ursprünglichen Versuchen mit der Software „Mamba“, konnte diesmal eine erhöhte Lautstärke des Ablenksystems bei erhöhter Scangeschwindigkeit festgestellt werden. Dies gilt jedoch nur für die normale Interpolation wie sie auch in der Software „Mamba“ verwendet wird. Beim neuen Ansatz erhöhte sich die Lautstärke des Systems nur minimal. Die messbare Erhöhung der Lautstärke im Gegensatz zum Ursprungsversuch, lässt sich damit erklären, dass das Ablenksystem bei der neuen Figur mehr Richtungswechsel und mehr Punkte abfahren musste, als im ursprünglichen Testframe.

6 ERGEBNIS / FAZIT

Die Evaluierung des neuen Interpolationsansatzes zeigte, dass diese maßgeblich dazu beitragen kann, bessere Projektionen ohne Korrekturmechanismen in der Software darzustellen. Zusätzlich ist es möglich, höhere Scangeschwindigkeiten zu erreichen, ohne das Ablenksystem unnötig zu belasten. Durch den geringeren Temperaturanstieg bei dem neuen Interpolationsansatz kann außerdem von einer längeren Lebensdauer des Ablenksystems ausgegangen werden. Die zunehmende Lautstärke des Ablenksystems bei hohen Scangeschwindigkeiten kann bei speziellen Installationen (beispielsweise in Museen) zum Problem

werden, da hier komplexe Projektionen und gleichzeitig hohe Scangeschwindigkeiten wünschenswert sein können.
Probleme könnten allerdings seitens des Laserschutzes auftreten aufgrund der helleren Stellen innerhalb der Projektion (vgl. [7] & [8]).
Weiterführende Forschung könnte eine Kombination aus beiden Interpolationsansätzen untersuchen, indem beispielsweise sichtbare Strecken innerhalb der Projektion (Laser an) mit einem normalen Interpolationsalgorithmus interpoliert werden, unsichtbare Stellen jedoch mit dem alternativen Interpolationsansatz.
Weiter könnte erforscht werden, wie groß die entsprechenden Zonen innerhalb einer Strecke zwischen Start- und Endpunkt sein müssen, damit die positiven Effekte des neuen Ansatzes feststellbar sind.

7 DANKSAGUNG

Hiermit danke ich der gesamten Firma Medialas für die Unterstützung und das zur Verfügung gestellte Material, insbesondere Herrn Dirk R. Baur und Hr. Björn Gerber.

8 LITERATURVERZEICHNIS

[1] Dirk R. Baur, Lasershow-Anlage Technik und Selbstbau, 3. Auflage, 1992, Elektor-Verlag, ISBN: 3-928051-24-5

[2] Dirk R. Baur, Das Laser Praxisbuch, 1997, Elektor-Verlag, ISBN: 3-928051-92-X

[3] Jürgen Eichler, Hans-Joachim Eichler, Laser: Bauformen, Strahlführung, Anwendungen, 2010, Gabler Wissenschaftsverlage

[4] MediaLas Electronics GmbH, FAQ Showlaser & Grundlagen, Oktober 2010

[5] MediaLas Electronics GmbH, User Manual / Benutzerhandbuch AttaXX Showlaser

[6] MediaLas Electronics GmbH, Mamba Laser Device (MLD) drivers, 5. Nov. 2006

[7] Unfallverhütungsvorschrift BGV B2 „Laserstrahlung“ von 1987 in der Fassung vom Januar 1997 mit Durchführungsanweisungen vom Oktober 1995 (Aktualisiert 2007)

[8] Deutsche Norm: DIN EN 60825-1: Sicherheit von Lasereinrichtungen – Teil1: Klassifizierung von Anlagen und Anforderungen (IEC 60825-1); Deutsche Fassung EN 60825-1

[9] International Laser Display Association, Technical Committee, ILDA Image Data Transfer Format, Revision 006, April 2004

[10] International Laser Display Association, Technical Committee, The ILDA Standard Projector, Revision 002, Juli 1999

[11] Coding4Fun Kinect Toolkit, c4kinect.codeplex.com, Einsichtnahme: 14.01.2012

[12] Kinect for Windows SDK Documentation, msdn.microsoft.com/en-us/library/hh855347.aspx, Einsichtnahme: 13.01.2012



Ermitteln des Netzwerkstatus in mobilen Web-Anwendungen

Marita Klein
Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Marita.Klein@Student.Reutlingen-University.de

ABSTRACT

Diese Arbeit betrachtet verschiedene Ansätze zur Erfassung des Netzwerk-Zustandes innerhalb des Browsers eines mobilen Gerätes. Zwei Ansätze werden näher betrachtet und miteinander verglichen. Mithilfe einer prototypischen Implementierung wurden Vergleichswerte ermittelt und ausgewertet. Auf Basis dieser Auswertung wird am Ende ein Fazit gezogen.

1 EINLEITUNG

Mobile Anwendungen zeichnen sich dadurch aus, dass sie auf tragbaren Geräten an allen möglichen Orten verwendet werden können. Dadurch kann ein durchgängiger Internetzugang nicht gewährleistet werden.
Bestimmte Anwendungen zeigen deshalb ein anderes Verhalten, wenn sie keinen Netzwerk-Zugriff haben. Beispielweise legt eine E-Mail Anwendung ohne Netzwerkzugang neue E-Mails erst einmal lokal an und sendet sie bei der nächsten Netzwerkverbindung ab. Hierfür ist es nötig den Netzwerk-Status zu kennen, um auf ein Offline-Verhalten ausweichen zu können.

2 MOBILE WEB-ANWENDUNGEN

Eine mobile Web-Anwendung positioniert sich als Anwendungsart zwischen einer Web-Anwendung und einer nativen mobilen Anwendung. Der Funktionsumfang wird vom Browser als Applikationscontainer bestimmt. HTML5 bietet einige Funktionalitäten, dank denen sich Webanwendungen ähnlicher zu nativen Anwendungen gestalten lassen. So lassen sich dank „Cache Manifest" und der lokalen Datenhaltung Anwendungen so gestalten, dass sie als eigenständige Anwendung auch offline ausführbar sind [1]. Da der Browser aber keinen Zugriff auf die Netzwerkkarte des Gerätes erhält, wird die Web-Anwendung hier eingeschränkt.

2.1 WEBBASIERTE NETZWERKKOMMUNIKATION

Die Netzwerkkommunikation innerhalb des Browser erfolgt über das HTTP-Protokoll. Dieses Positioniert sich im OSI-Modell in der Anwendungsschicht. Eine bestehende TCP-Verbindung (Transportschicht) ist für alle Übertragungen in der Anwendungsschicht notwendig. Diese wird durch einen 3-Wege-Hanschlag initiiert: (SYN, SYN/ACK, ACK). Das HTTP-Protokoll dient dem Austausch von Daten zwischen Browser und Web-Server (Client-Server-Protokoll). Eine HTTP-Nachricht besteht deshalb grundsätzlich aus einer Anfrage und einer Antwort. (vgl. [2], S.21ff)

2.2 WEBSOCKETS

Die HTML5-Spezifikation stellt eine API für einen bidirektionalen, full-duplex Kommunikationskanal auf Basis von TCP-Sockets zwischen Server und Client bereit. Die WebSocket API [7] sieht die Möglichkeit vor, dass der Client eine Verbindung zu einem Server öffnen und über diese Verbindung Daten sowohl senden als auch empfangen kann. Im Unterschied zum HTTP-Protocol, ist nach dem Verbindungsaufbau ein permanentes Senden und Empfangen von Daten ohne erneuten Handschlag und ohne HTTP-Header möglich (vgl. [11], S.159f).

3 ANSÄTZE

Um den Netzwerkstatus innerhalb des Browsers zu erfahren, gibt es 3 Ansätze [5]. Zudem wurde eine eigene Idee ausgearbeitet.

3.1 XMLHTTPREQUEST

Mithilfe von XHR, oder auch Ajax, wird einem Browser das Senden und Empfangen von Daten eines Web-Servers ohne das Neuladen der Seite ermöglicht. Hierfür sind das dahinterstehende HTTP und dessen Netzwerktechniken identisch wie beim Seitenaufbau. Angefragt wird aber ein JavaScript-Objekt, sodass die Antwort des Servers als asynchrone Rückruffunktion (Callback) verarbeitet werden kann.
Ein Fehler in der Anfrage löst ein Event aus. Mithilfe dieses Fehlers und der entsprechenden Fehlermeldung lässt sich auf die Netzwerk-Verbindung schließen [6].
Um den Netzwerk-Zustand aktuell zu halten, kann man periodisch auf eine Ressource auf einem Server zugreifen.

3.2 HTML5-FLAG

Bei ersten Recherchen zum Thema Online-Status von Webanwendungen stößt man auf das HTML5-Flag „window.navigator.onLine“. Es zeigt den Status der Internetverbindung an ([5][8]). Gleichfalls gibt es ein Event, dass auf Änderungen dieses Flags hört.
Das Verhalten dieses Flags zum Zeitpunkt dieser Ausarbeitung ausschließlich im Android Browser ab Version 2.2 implementiert [9]. In allen anderen Browsern zeigt es ein nicht standardisiertes Verhalten. Einige Tests kamen zu dem Ergebnis, dass beispielsweise das Flag unter Chrome dann geändert wurde, wenn ein XMLHttpRequest erfolglos war. Demnach weiß man den Online-Status der Anwendung erst im Anschluss an eine Anfrage, nicht aber vorher, wie es nötig wäre. Ein anderer Test mit dem iOS-Simulator, bei dem das Netzwerk des Computers abgeschaltet war, zeigte permanent einen Online-Status an.

3.3 APPCACHE FEHLER EVENT

In Kapitel 2 wurde das Cache Manifest erwähnt. Dieses Manifest enthält eine Liste aller Ressourcen, die offline verfügbar gemacht werden sollen. Der Browser prüft in regelmäßigen Abständen, ob Änderungen dieser Datei auf dem Server Vorgenommen wurden, um die Ressourcen bei Bedarf zu aktualisieren [4].
Dieses Verhalten lässt sich aber auch nutzen, um die Netzwerkkonnektivität zu prüfen. Schlägt die Anfrage auf das Manifest fehl und dessen Existenz kann Server-seitig weiterhin garantiert werden, liegt höchstwahrscheinlich ein Netzwerk-Problem vor. Das sogenannte AppCache-Event hört auf diese Fehler und kann somit auch den Bedürfnissen entsprechend implementiert werden.

3.4 WEBSOCKET-VERBINDUNG

Ähnlich dem XMLHttpRequest-Ansatz kann mit einer WebSocket-Verbindung gezielt oder periodisch durch das Senden und Empfangen einer Nachricht der Netzwerk-Zustand geprüft werden.

4 EVALUIERUNG DER ANSÄTZE

Betrachtet man das AppCache-Event bietet dieser Ansatz eine intuitive Verwendung eines Netzwerk-Zugriffs, der sowieso ausgeführt wird. Bei näherer Betrachtung ist diese Verwendung für eine Client-seitige Netzwerk-Prüfung aber eher ungeeignet. Grund ist das Intervall des Zugriffs. Die HTTP-Anfrage auf das Cache Manifest erhält zur Antwort das Manifest (oder die Meldung „Not Modified“) und ein Datum bis zu diesem das Manifest gültig ist. Dieses Datum steuert direkt den nächsten Zugriff und wird daher von Seiten des Servers initiiert. Eine weitere Schwäche zeigt sich durch die Fehlerbehandlung. Nicht notwendigerweise lässt ein Fehler im Abruf des Manifests auf das Nichtvorhandensein der Internetverbindung schließen. Beispielsweise könnte das Manifest unter der angegebenen URI nicht mehr existieren.
Das Flag „window.navigator.onLine“ ist, wie schon in 3.2 beschrieben, bislang in sehr wenigen Browsern implementiert und bietet daher keine ausreichende Zuverlässigkeit. Ein weiteres Problem stellt die Unterscheidung von Netzwerk-Zugang und Internet-Zugang dar. Wenn das Gerät erfolgreich im lokalen W-



LAN-Netzwerk verbunden ist, kann es trotzdem sein, dass der Router keine Verbindung nach außen hat.
Der Ansatz über Ajax-Anfragen den Zustand zu prüfen erschien bei den Tests am Zuverlässigsten. Probleme zeigten sich aber gerade in Browsern mobiler Geräte. Wenn eine HTTP-Anfrage mit der Methode „GET“ an eine identische URI gesendet wird, speichert der Browser das Ergebnis ab und gibt bei neuen Anfragen dieses Ergebnis zurück. Dies ließe sich durch eine Hilfskonstruktion umgehen, ein weiterer nicht unerheblicher Nachteil zeigt sich aber im Netzwerkverkehr, der durch ständiges Überprüfen entsteht. Aus diesem Grund wurde diese Idee auf WebSockets übertragen, die dank einmaliger Verbindung den Daten-Overhead, aber auch die Latenz-Zeit minimieren. Nachfolgend wird daher dieser Ansatz herausgearbeitet und implementiert. In Kapitel 7 sieht man anschließend den direkten Vergleich.

5 ANFORDERUNGEN

Die Anforderungen an eine Netzwerkprüfung resultieren aus dem Anwendungsgebiet. Hauptaugenmerk wurde im Rahmen dieser Ausarbeitung auf zwei Kriterien gelegt. Zum einen ist es wichtig, dass die Netzwerk-Prüfung einen möglichst kleinen Overhead verursacht. Es darf so wenig unnötiger Datenverkehr wie möglich aufkommen. Dies hat zwei Gründe. Erstens haben mobile Geräte meist einen Volumenbegrenzten Internetzugang, zweitens ist der Netzzugang häufig sehr langsam. Dies kann zu einer schlechten Performance führen, wenn große Datenmengen übertragen werden. Die zweitwichtigste Anforderung ist die Akkulaufzeit des mobilen Endgerätes. Die Implementierung sollte wenig Einfluss auf die Akkuleistung haben, und daher möglichst geringen Rechenaufwand innehaben. (vgl: [3]) Diese beiden Kriterien wurden mithilfe von Benchmarks anhand einer prototypischen Implementierung überprüft.

6 IMPLEMENTIERUNG

Implementiert wurden zwei unterschiedliche Ansätze. Einmal ein periodisches Pingen, dass ein permanentes gewahr sein des Netzwerkstatus erlaubt, sowie auch Informationen zur Qualität der Netzwerkverbindung anhand der Latenz liefern kann. Außerdem wurde ein sporadisches Pingen implementiert, dass vor einem gewünschten Request ausgeführt werden kann. Im Offline-Fall kann die geplante Anfrage gespeichert und beim nächsten Verbindungsaufbau durchgeführt werden.
In beiden Fällen wird im Offline-Fall ein Wiederverbinden versucht. Dies geschieht im Falle eines Nichterfolgs in immer größeren Abständen, um den Akku des Gerätes zu schonen.

7 BENCHMARKS

Um die Erfüllung der Anforderungen an ein On-/Offline-Awareness System zu überprüfen, werden in diesem Abschnitt anhand einiger Zahlenbeispiele Hochrechnungen durchgeführt.

7.1 OVERHEAD

Mithilfe des Netzwerk-Sniffers Wireshark[1] wurden Durchschnittswerte erfasst, die zur Hochrechnung des Overheads verwendet wurden. Neben den in Tabelle 1 aufgelisteten Werten, gilt zu jeder HTTP-Nachricht noch ein TCP-Ack hinzuzurechnen. Für jedes Empfangen einer HTTP-Nachricht antwortet der Empfänger mit einem ACK um den Empfang der Daten zu bestätigen. Da diese Werte aber für WebSockets und Ajax-Anfragen identisch sind wurden sie nicht aufgeführt.
Da für die Implementierung die Bibliothek Socket.IO [10] verwendet wurde, entstehen durch

Tabelle 1: Vergleich Daten-Volumen WebSocket mit XHR

	XHR	WebSocket
Verbindungsaufbau/Reconnect	222 Byte (TCP)	1902 Byte
Einmaliges Prüfen	753 Byte	290 Byte
Verbindungsabbau (Timeout)	330 Byte (TCP)	330 Byte (TCP)

[1] „the worlds foremost network protocol analyzer“ http://www.wireshark.org/

dort implementierte Automatismen auch einige Nebeneffekte. In diesem Fall ist der Nachteil, dass der HTTP-Overhead von 2 Byte (WebSocket Protokoll) durch ein eigenes Socket.IO-Protokoll auf 25 Byte erhöht wird.
Für das sporadische Testen mit den obigen Ergebnissen muss die WebSocket-Verbindung bestehen bleiben. Socket.IO ist hierfür mit einem maximalen Timeout von zwei Minuten implementiert. Dies bedeutet für die sporadischen Tests, dass mindestens alle zwei Minuten eine Anfrage versendet werden muss. Wird kein Test in dieser Zeit initiiert, wird ein „keep-alive“-Ping versendet. So werden die im Verhältnis großen Verbindungs- und Abbau-Netzwerklasten vermieden.
Die Ergebnisse der verschiedenen Testarten zeigt Abbildung 1. Für das periodische Testen wurde eine Stunde lang jede Sekunde ein Ping versendet. Für das sporadische Testen wurde innerhalb einer Stunde zufällig 25 Mal ein Ping vom Client aus abgesetzt.

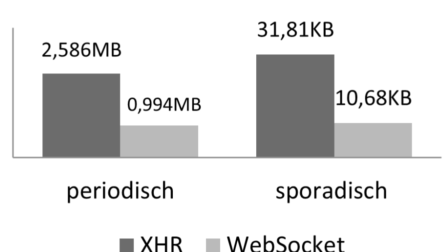


Abbildung 1: Daten-Volumen Vergleich

Wichtig bei der Auswertung der Berechnungen ist die Stabilität des Netzwerkes. Der Verbindungsaufbau der WebSockets ist deutlich höher als bei einem Ajax-Aufruf. Bricht also beispielsweise beim sporadischen Testen die Verbindung 11 mal ab, ist das Daten-Volumen wieder gleich hoch wie beim XHR-Aufruf.

7.2 AKKULAUFZEIT

Ein nichtrepräsentativer Test unter Laborbedingungen hat ergeben, dass zwischen periodischem Pingen via WebSocket sowie Ajax kein signifikanter Unterschied im Akkuverbrauch festzustellen ist. Für den Test wurde ein BlackBerry Playbook verwendet. Es gab sechs Testfälle, je drei für eine halbe Stunde und je drei für eine Stunde. Vor jedem Test wurde der Akku des Geräts zu 100% aufgeladen und während des Tests war das Display stets beleuchtet. Die drei Testszenarien waren Leerlauf im Browser, also keine JavaScript-Ausführungen im Hintergrund als Vergleichswert, WebSocket-Polling und Ajax-Polling.

Tabelle 2: Akkulaufzeiten

	Leerlauf	Ajax	WebSocket
1/2 Stunde	6%	9%	10%
1 Stunde	11%	20%	21%

Bei Polling wurde jeweils einmal pro Sekunde eine Anfrage gesendet. Am Ende jedes Testes wurde erneut der Akku-Status abgelesen und die Differenz berechnet. Erfasst wurde also der Akkuverbrauch in den jeweiligen Zeiträumen. Die Ergebnisse zeigt Tabelle 2.

8 FAZIT

Beim periodischen Pingen konnte durch die Verwendung von WebSockets die Netzwerk-Last um den Faktor 2,6 verringert werden. Bei sporadischem Testen sogar um den Faktor 3. Beim sporadischen Prüfen entsteht dieser höhere Wert aus dem häufigen Ab- und Wiederaufbau der TCP-Verbindung bei einfachen HTTP Anfragen.
Ist jedoch mit häufigen Netz-Abbrüchen zu rechnen, relativiert sich dieser Vorteil wieder, da die Datenübertragung beim Verbindungsaufbau um 8,5 Mal größer ist als bei XHR.
WebSockets erwiesen sich aber gerade beim periodischen Ping als leichtgewichtigste und auch schnelle Methode. Eine Studie des Ericsson Labs zur Echtzeitfähigkeit bestätigt dies, WebSockets seien im Vergleich zu XHR drei bis fünf Mal schneller [12].
Beim periodischen Pingen ist aber generell Vorsicht geboten. Es sind zwar keine signifikanten Unterschiede zwischen dem Akkuverbrauch von WebSocket und Ajax-Anfragen

festzustellen, der Akkuverbrauch stieg durch die Netzwerklast aber um fast 50% an. Gleichzeitig werden im besten Fall rund 1MB an Overhead an Daten pro Stunde übertragen. Beide Werte müssen in Zusammenhang mit den Anforderungen aus Kapitel 4 betrachtet werden. Da mobile Anwendungen aber unter Umständen nur für sehr kurzweilige Aufgaben verwendet werden, kann keine pauschale Bewertung der erfassten Werte vorgenommen werden. Je nach Anwendungsfall gilt es daher, eine entsprechende Gewichtung dieser und ggf. weiterer Anforderungen vorzunehmen.

9 LITERATURVERZEICHNIS

[1] Anne van Kersteren Ian Hickson. Offline Web Applications.www.w3.org/TR/offline-webapps/, 2011. Einsichtnahme: 20.12.2011.

[2] Albrecht Becker. Jetzt lerne ich TCP/IP. Mark + Technik, 2. Auflage, 2003

[3] Oriana Riva and Jaakko Kangasharju. 2008. Challenges and Lessons in Developing Middleware on Smart Phones. Computer 41, 10 (October 2008), 23-31.

[4] Ian Hickson. Application cache API. http://www.w3.org/TR/html5/offline.html#application-cache-api, Einsichtnahme: 09.12.2011

[5] HTML5 ROCKS. Working Off the Grid with HTML5 Offline. http://www.html5rocks.com/en/mobile/workingoffthegrid.html#toc-detecting-network, 2011. Einsichtnahme: 09.12.2011.

[6] Anne van Kesteren. The XMLHttpRequest Object. http://www.w3.org/TR/2007/WD-XMLHttpRequest-20070227/, Einsichtnahme: 09.12.2011

[7] Ian Hickson. The WebSocket API. http://dev.w3.org/html5/websockets/, Einsichtnahme: 16.12.2011

[8] Suresh Chitturi, Robin Berjon. Network Information API. http://www.w3.org/TR/netinfo-api/. Einsichtnahme: 20.01.2012

[9] Mobile HTML5. Compatibility on mobile and tablet Browsers. http://mobilehtml5.org/, Einsichtnahme: 12.12.2011

[10] Guillermo Rauch. Socket.IO https://github.com/LearnBoost/sockt.io, 2011. Einsichtnahme: 15.12.2011.

[11] Brian Albers Peter Lubbers, Frank Salim. Pro Html5 Programming: Powerful Apis for Richer Internet Application Development. Apress, 2011. Second Edition.

[12] Ericsson Lab, Web Connectivity, 2010, https://labs.ericsson.com/apis/web-conectivity/documentation, Einsichtnahme: 17.02.2012



Persistenz von EMF-Modellen

Oliver Schumann
Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Oliver.Schumann@Student.Reutlingen-University.de

ABSTRACT

Im Rahmen der wissenschaftlichen Vertiefung, an der Hochschule Reutlingen, geht diese Arbeit auf die Persistenz von Modellen des EMF (Eclipse Modeling Framework) ein. Zu Beginn geht diese Arbeit auf die Grundlagen von EMF, die Persistenz des Frameworks und deren Komplikationen im Rahmen größerer Modelle ein. Diese Arbeit stellt Teneo, CDO (Connected Data Objects) und EMFStore als Lösungsansätze zur Persistenz von Modellen für große Projekte vor. Zu diesem Zweck wurden Prototypen für die einzelnen Lösungsansätze erstellt.

1 EINLEITUNG

Erstellen von Modellen zur Dokumentation von Softwaresystemen oder gar zur Codegenerierung gewinnen zunehmend an Bedeutung in der Softwareentwicklung. Modelle sind nach H. Stachowiak [1] ein beschränktes Abbild der Wirklichkeit und sind durch drei Merkmale gekennzeichnet:

- Abbildungsmerkmal
- Verkürzungsmerkmal
- Pragmatisches Merkmal

In der Informatik sind Modelle meist Darstellungen in Form von Sprache. Dies können geschriebene oder gesprochene Texte, Bilder oder Grafiken sein. Sie verweisen auf das, was repräsentiert werden soll und stellen somit eine Abstraktion des Modellierten dar [2].

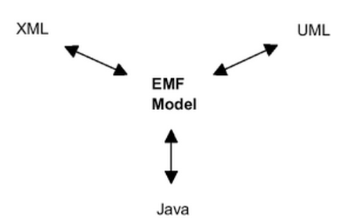


Abbildung 1: EMF vereinigt UML, XML und Java [3]

So ist es möglich Modelle zum Beispiel in der Modellierungssprache UML[1], als XML[2] Schema oder als Interfaces der Programmiersprache Java zu beschreiben. EMF vereinigt diese Technologien miteinander, wie Abbildung 1 darstellt.

2 EMF UND SEINE PERSISTENZ

EMF ist ein Framework, das aus den Grundkonzepten der Metadaten, Codegenerierung und Serialisierung basiert. Es ist ein Teil aus dem Eclipse Modeling Project und dient dazu die Implementierung und das Design eines strukturierten Modells zu vereinfachen. Weiterhin basiert es auf der Programmiersprache Java. EMF legt den Schwerpunkt nicht auf programmiertechnische Details, sondern konzentriert sich auf die Modellierung. [3]
Unabhängig davon welche Technologie verwendet wird um ein Modell zu erstellen, kann EMF aus dieser Beschreibung die Modelle der jeweils anderen Technologie generieren.
Für die Definition eines Modells in EMF wurde ein eigenes Metamodell kreiert. Es heißt Ecore und ist selbst als EMF-Modell beschrieben und ebenfalls sein eigenes Metamodell. Es gibt vier



Abbildung 2: UML-Diagramm [3]

Ecore-Klassen die benötigt werden um ein Modell darstellen zu können.

- EClass wird verwendet um Klassen zu modellieren. Jede Klasse hat einen Namen und kann Attribute sowie Referenzen besitzen
- EAttribute ist das Konzept um Attribute zu modellieren. Attribute haben einen Namen und einen Typ.

[1] Unified Modeling Language
[2] eXtended Markup Language

- EReference wird verwendet um ein Ende einer Assoziation (Beziehung) zwischen Klassen darzustellen, sie besitzt einen Namen, ein boolesches Flag um anzugeben ob es Containment[3] darstellt und ein Zieldatentyp, der die Klasse am anderen Ende bezeichnet
- EDataType stellt den Datentyp eines Attributes dar, der sowohl Primitiv- als auch Objektdatentyp sein kann

Die Notationen von Ecore ähneln der Notation von UML. Dies liegt daran, dass Ecore eine kleine und vereinfachte Teilmenge der UML ist.
EMF besitzt einen einfachen und doch mächtigen Mechanismus um Objekte zu persistieren. Die erzeugten EMF-Objekte werden über einen Serialisierungsmechanismus gespeichert. In EMF sind zwei Serialisierer implementiert: XML und XMI[4]. Dieser Serialisierungsmechanismus ist als offene Schnittstelle definiert und bietet die Möglichkeit der Erweiterung.

```
1. <?xml version="1.0" encoding="UTF-8">
2. <po:PurchaseOrder xmi:version="2.0"
     xmlns:xmi="http://ww.omg.org/XMI"
     xmlns:po="http://
       www.example.com/simplePO"
     billto="123 Maple Street">
3. <items productName="Apples" quantity="12"
     price="0.5" />
4. </po:PurchaseOrder>
```

Codebeispiel 1: mypo.xml [3]

Diese XML-Daten werden durch EMF auf der Festplatte als ASCII- oder Unicode-Datei gespeichert. Codebeispiel 1 zeigt ein solches XMI-Dokument.
Das boolesche Flag Containment in der Ecore-Klasse EReference ist ein Attribut logischer Beziehungen. Nutzt man eine XML-Ressource um ein Modell zu speichern, landen alle Objekte in einer Datei, die vom Wurzelelement durch das Containment verbunden sind. Da dateibasierte Speicherung keinen wahlfreien Zugriff einzelner Abschnitte zulässt, muss sie komplett gelesen bzw. geschrieben werden. Ersetzt man die logischen Containments durch Cross References, wird eine große Datei in kleinere Dateien zerteilt. Dies ist jedoch ein Mehraufwand an der Benutzerschnittstelle und die logische Objektstruktur wird im Speicher dennoch wieder zusammengeführt.
Dieser Serialisierungsmechanismus zur Persistierung von Modellen bietet sich daher nur für kleine Anwendungen an und nicht für größere Datenmengen.

3 DATENBANKBASIERTE LÖSUNGSANSÄTZE

Im Folgenden werden die verschiedenen Datenbankansätze Teneo/Hibernate, CDO-Model Repository sowie EMFStore vorgestellt.

3.1 TENEO/HIBERNATE

Teneo bietet eine Möglichkeit EMF-Modelle in einer relationalen Datenbank zu speichern. Abbildung 3 zeigt das grundlegende Verfahren von EMF-Modellen zur Laufzeit von Teneo. Der Ausgangspunkt ist ein EMF-Modell, das durch eine oder mehrere EPackages vertreten ist, welche durch das Metamodell Ecore beschrieben sind. Diese EPackages können EJB3[5] ähnliche Annotationen enthalten. [5]

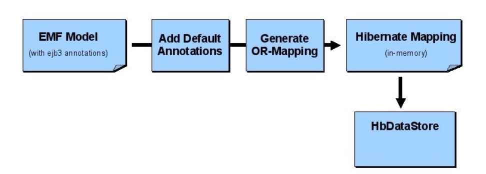


Abbildung 3: Teneo Mapping Übersicht [5]

Nach dem Start der Clientanwendung werden die EPackages in Teneo registriert. Die registrierten EPackages werden zu einem im Speicher annotierten Modell übersetzt. Dieses annotierte Modell enthält zunächst nur die

[3] Containment ist die Bezeichnung für den Beziehungstyp der Komposition („besteht aus"). Klasse „A" bildet den Container für das Objekt der Klasse „B", welches ohne Klasse „A" nicht existieren kann.

[4] XMI steht für XML Metadata Interchange. Es ist ein XML basierendes Standardformat der Object Managment Group (OMG) und bildet somit die Grundlage für die Interoperabilität zwischen verschiedenen Werkzeugen.

[5] EJB3 kommt zum Einsatz wenn das automatische Verhalten nicht dem erwünschten Verhalten entspricht. Diese Notationen werden nicht überschrieben.

Annotationen, die manuell angegeben wurden als EAnnotations im ursprünglichen Modell.
Im nächsten Schritt fügt Teneo automatisch Annotationen hinzu um das EMF-Modell in einer Darstellung abzubilden. Die manuell eingegebenen EJB3-Annotations werden beibehalten und nicht überschrieben.
Das Modell ist nun vollständig annotiert und wird in einen „hibernate.hbm.xml"-String übersetzt, welche die Mapping-Informationen enthält. Dieses Hibernate Mapping wird dann verwendet, um den HbDataStore zu konfigurieren.

3.1.1 RUNTIME LAYER

Die wichtigste Komponente der Runtime Layer ist der HbDataStore, siehe Abbildung 4.
Der HbDataStore kontrolliert eine SessionFactory und eine Reihe von EPackages. Diese werden innerhalb der Session gespeichert, welche von der Session-Factory zugewiesen wurde. Während der Verwendung einer Hibernate-Session, kümmert sich Teneo, mit dem EMF-Interceptor, hinter den Kulissen um die Instanziierung der EMF-Objekte.

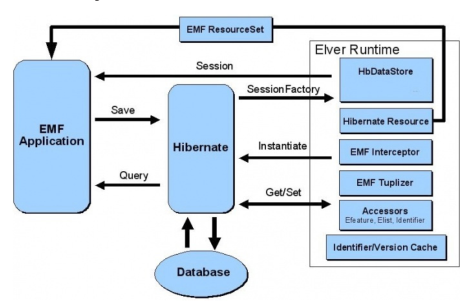


Abbildung 4: Runtime Layer [5]

Das Setzen und Laden der einzelnen Eigenschaften der einzelnen Objekte aus der Datenbank übernehmen die Accessors. Das Framework Hibernate sorgt als O/R-Mapper dafür, dass die objektorientierten Daten von EMF in einer relationalen Datenbank gespeichert werden.

3.2 CDO MODEL REPOSITORY

CDO ist ein reines Java-Modell-Repository für EMF-Modelle und Meta-Modelle. Es unterstützt eine 3-Tier-Architektur, wie Abbildung 5 zeigt. Auf der linken Seite der Abbildung sind die EMF-basierten Client-Anwendungen dargestellt. Diese sind mit dem zentralen CDO-Model-Repository verbunden. Zur Datenspeicherung können verschiedene Back-Ends verwendet werden. Dies können unter anderem relationale Datenbanken, Objektdatenbanken oder NoSQL-Datenbanken sein. Denkbar sind auch Dateisysteme, Hibernate oder Webservices. Ist das Model-Repository entsprechend konfiguriert, ist auch eine Versionierung des Objektbaumes möglich, falls das Backend hierfür eine entsprechende Implementierung bietet.

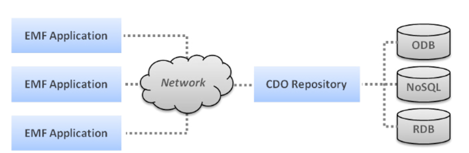


Abbildung 5: CDO Übersicht [7]

CDO hält den Anwendungscode frei von herstellerspezifischen Daten-Zugriffcodes und erleichtert die Übergänge zwischen diesen unterstützten Backend-Typen.
Dieses Framework hat eine hohe Skalierbarkeit. Objekte werden erst bei Bedarf geladen und nicht mehr referenzierte Objekte werden dem Garbage-Collector zugeführt. Weiterhin ist eine Multi-User-Unterstützung gegeben, indem lokale Objekte durch eine ständige Verbindung zum Model-Repository-Server permanent aktualisiert werden. Somit arbeitet jeder Nutzer auf dem aktuellen Modell.
[10]
Die Architektur des CDO umfasst Applikationen und Repositories. Trotz einer Reihe von eingebetteten Optionen werden Anwendungen in der Regel zu Clients und Repositories zu Servern. Sie kommunizieren über ein für CDO entwickeltes Protokoll auf der Anwendungsebene. [7]

3.2.1 CLIENT ACHITEKTUR

Die Architektur einer CDO-Anwendung zeichnet sich durch ihre verbindliche Abhängigkeit von EMF aus. Die meiste Zeit interagiert die Anwendung mit dem Objektdiagramm des Modells über Standard-APIs von EMF, da CDO-Modellobjekte von EMF-Objekten abgeleitet sind.
Die Grundfunktion von CDO integriert sich perfekt und transparent mit dem EMF-Erweite-

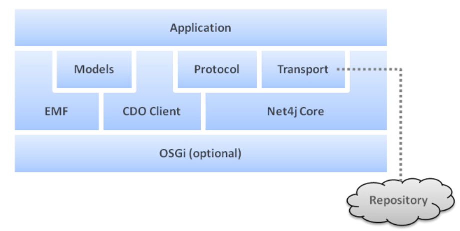


Abbildung 6: Client Architektur [6]

rungsmechanismus. Die zusätzlichen erweiterten Funktionen haben direkte Abhängigkeit zu CDO und diese müssen im Programmcode hinzugefügt werden. Abbildung 6 zeigt die wichtigsten Bausteine einer CDO-Anwendung.

3.2.2 REPOSITORY ARCHITEKTUR

Der Hauptbestandteil eines CDO-Repositories ist in zwei Ebenen aufgeteilt - eine allgemeine Ebene, mit der Applikationen interagieren, und eine Datenbankebene, in die Provider ihre Datenspeicherlösungen integrieren können. Eine Reihe solcher Integrationen werden bereits mit CDO mitgeliefert, die es ermöglichen, ein Repository für alle Arten von JDBC-Datenbanken zu verbinden. [8]
Während technisch ein CDO-Repository von EMF abhängig ist, hat diese Abhängigkeit nicht die gleiche Bedeutung, wie in einer CDO-Anwendung. Insbesondere die generierten Anwendungsmodelle müssen nicht auf dem Server bereitgestellt werden, weil ein CDO-Repository auf Modelle reflektiv zugreift und die Modell-Objekte werden nicht als EObjects auf dem Server implementiert.
Abbildung 7 zeigt die wichtigsten Bausteine eines CDO-Repositories.

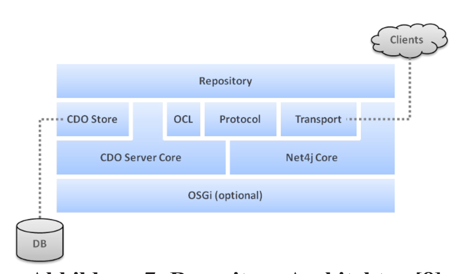


Abbildung 7: Repository Architektur [8]

3.3 EMFSTORE

EMFStore entstand an der Technischen Universität München im Rahmen des UNICASE-Projektes. UNICASE ist ein uniformes CASE-Tool für die Softwareentwicklung.
EMFStore ist ein operationsbasiertes System zur Versionskontrolle für Modelle, welche auf EMF basieren. Es ermöglicht mit Modellen kollaborativ zu arbeiten, zu speichern und zu versionieren. Zu seinem Funktionsumfang gehören Änderungsverfolgung, Konflikterkennung, Versionierung und Zusammenführen von Modellen. Es besteht aus einem Server und einem Client, wie in Abbildung 8 zu sehen ist.

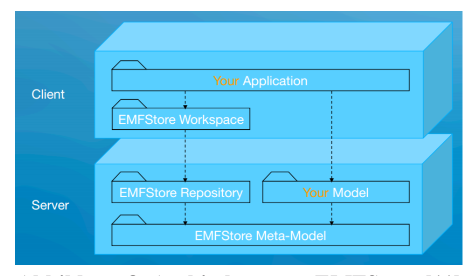


Abbildung 8: Architektur von EMFStore [11]

Der Server läuft als eigenständige Komponente und bietet ein Repository für Modelle mit Versionierung, Persistenz und Zugriffskontrolle. Die Clientkomponente ist gewöhnlich in der Anwendung integriert. Die Aufgaben dieser Komponente besteht aus folgenden Funktionen:

- Change Tracking: Änderungen am Modell werden aufgezeichnet.
- Commit: Die lokalen Änderungen an Modellen können an das Repository gesendet werden.
- Update: Änderungen aus dem Repository werden an den lokalen Arbeitsbereich gesendet.
- Checkout: Um Modelle offline zu bearbeiten, werden sie auf den lokalen Arbeitsbereich übermittelt.
- Merging: Zusammenführen von verschiedenen Versionen der Modelle.

EMFStore arbeitet nicht wie herkömmliche Versionierungssysteme, wie Subversion oder CSV,

welche nur den Zustand eines Modells speichern.
Jede Änderung am Modell wird aufgezeichnet und diese Änderungsoperationen werden bis zum nächsten Commit ins Repository gespeichert. Die Änderungen welche am Modell vorgenommen werden unterliegen gewissen Einschränkungen, so können z.B. Änderungen nur über die vorgeschriebenen Methoden verändert werden und nicht über manuelle Eingriffe in den Modell-Dateien. Durch diese Art der Aufzeichnung kann jeder Zustand des Modells rekonstruiert werden und eine feingranulierte Konflikterkennung ist beim Merging dadurch möglich. Ein manueller Eingriff beim Merging ist nur notwendig, wenn zwei Operationen im Widerspruch stehen. Dies könnte sein, wenn zwei Operationen, die gleichen Werte im gleichen Attribut des gleichen Modells zu einem Element ändern und sich die Ergebnisse voneinander unterscheiden. [11]

4 FAZIT

Für kleinere Projekte ist die Standardmethode, in diesem Fall XML-Serialisierung, zur Speicherung für EMF-Modelle geeignet. Bei umfangreicheren Projekten stößt diese Methode an ihre Grenzen. Alle der hier vorgestellten Frameworks bieten eine gute Alternative zur Persistenz von EMF. Teneo und CDO können mit verschiedenen Datenbanken kombiniert werden, sodass sie in eine bereits vorhandene Infrastruktur integriert werden können. EMFStore nutzt seine eigene Implementierung um die Modelle zu persistieren und kann derzeit nicht mit anderen Datenbanksystemen kombiniert werden. Letztendlich bietet CDO durch sein größeres Spektrum an Funktionen die bessere Lösung zur Persistenz und eine höhere Skalierbarkeit.

5 LITERATURVERZEICHNIS

[1] Stachowiak, H. 1973. Allgemeine Modelltheorie. Springer-Verlag Wien

[2] Hesse, W.; Mayr, H. C. 2008. Modellierung in der Softwaretechnik: eine Bestandsaufnahme. Informatik-Spektrum 31 Nr. 5. S. 377–393

[3] Steinberg, D; Budinsky, F.; Paternostro M.; Merks E. 2009. EMF: Eclipse Modeling Framework. Pearson Education Boston

[4] Bachman, J. 2007. Entwurf und Implementierung eines graphischen Modelleditors und einer Benutzerschnittstelle für das Werkzeug CASPA. Universität München http://www.unibw.de/inf3/personen/wimis/riedl/studarb/abgeschlossene-arbeiten/diplomarbeit_bachmann.pdf. Letzter Zugriff: 03.03.2012

[5] Teneo/Hibernate/Architecture Overview. http://wiki.eclipse.org/Teneo/Hibernate/Architecture_Overview#Teneo_Mapping_Overview. Letzter Zugriff: 03.03.2012

[6] Stepper, E. Understanding the Architecture of a Client Application. http://download.eclipse.org/modeling/emf/cdo/drops/I20120217-0410/help/org.eclipse.emf.cdo.doc/html/programmers/client/Architecture.html. Letzter Zugriff: 03.03.2012

[7] Stepper, E. CDO Model Repository Overview. http://www.eclipse.org/cdo/documentation/. Letzter Zugriff: 03.03.2012

[8] Stepper, E. Understanding the Architecture of a Repository. http://download.eclipse.org/modeling/emf/cdo/drops/I20120217-0410/help/org.eclipse.emf.cdo.doc/html/programmers/server/Architecture.html. Letzter Zugriff: 03.03.2012

[9] Moore B.; Dean, D.; Gerber, A.; Wagenknecht, G.; Vanderheyden. 2004. Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework. http://www.redbooks.ibm.com/redbooks/pdfs/sg246302.pdf. Letzter Zugriff: 03.03.2012

[10] Stepper, E. 2009. Modeling goes Enterprise. JAXenter http://it-republik.de/jaxenter/artikel/Modeling-goes-Enterprise-2503.html Letzter Zugriff: 03.03.2012

[11] Koegel, M.; Helmig J. 2010 EMFStore - a Model Repository for EMF models http://www1.informatik.tu-muenchen.de/static/publications/pdf/208/Paper1.pdf. Letzter Zugriff: 03.03.2012

Regelbasierte Erstellung von Sicken zur Gewichtsreduktion von Montageplatten

Christopher Mildner
Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Christopher.Mildner@Student.Reutlingen-University.de

ABSTRACT

Im Rahmen der wissenschaftlichen Vertiefung behandelt dieser Artikel die Entwicklung eines Algorithmus zur Erstellung von Versteifungssicken, welche in der Luft- und Raumfahrttechnik zur Gewichtsreduktion von Montageplatten eingesetzt werden. Für die automatische Generierung dieser Sicken wird eine graphenbasierte Entwurfssprache erstellt, die basierend auf wenigen Parametern durch einen Compiler in ein Geometriemodell (vergleichbar mit einem CAD[1]-Modell) übersetzt wird.

1 EINLEITUNG

Im Kontext der Luft- und Raumfahrt werden fortwährend Möglichkeiten gesucht, um Masse und damit letztendlich auch Energie einzusparen. Eine zweckmäßige Möglichkeit ist es, aus Einsparungsgründen dünneres und somit gewichtreduziertes Material zu verbauen. Natürlich ist dabei trotz der Gewichtsersparnis die notwendige Stabilität des benötigten Bauteils zu gewährleisten, da z.B. beim Transport eines Satelliten ins Weltall starke Kräfte auf diesen einwirken. Um die erforderliche Materialbeständigkeit und dadurch eine stabile Konstruktion zu erreichen, werden sogenannte Sicken eingesetzt. Am Institut für Statik und Dynamik der Luft- und Raumfahrtkonstruktionen (ISD) an der Universität Stuttgart können Satellitenkonstruktionen in einer eigens dazu entwickelten, graphenbasierten Entwurfssprache generiert werden. Mittels dieser Entwurfssprache wird die Konzeption komplexer Systeme systematisiert [1]. Dazu wird der Entwurf in Vokabeln zerlegt und über Regeln miteinander verbunden [2]. So werden mit Entwurfssprachen am ISD beispielsweise Satelliten, Flugzeuge oder Strommasten entworfen. Spezielle für die Entwurfssprache verfügbare Interfaces (Plug-Ins für Geometrie, Regelung, …) erlauben die Visualisierung eines Satellitenentwurfs in unterschiedlichen Simulationsumgebungen.

Ein über die Entwurfssprache beschriebenes Modell kann von einem Compiler (d.h. Übersetzer) in ein CAD-Modell übersetzt werden. Bei dem Compiler handelt es sich um den von der IILS[2] mbH in Kooperation mit der Universität Stuttgart entwickelten „Design Compiler 43". Dabei ist es möglich auf die häufigsten CAD-Geometrieformen wie Kreise, Rechtecke, Zylinder, etc. zurückzugreifen. Gleichzeitig können weitere Modellsprachen, wie mathematische Gleichungen und Finite-Elemente Methoden, genutzt werden.

2 SICKEN

Als sogenannte „Sicke" werden Vertiefungen oder Erhöhungen bezeichnet, welche senkrecht zur Oberfläche einer Leitstützstruktur angebracht sind um deren Festigkeit zu erhöhen [3]. In der Geometrie werden diverse Varianten von Sicken unterschieden. Am häufigsten kommen die in Abbildung 1 dargestellten Sickenprofile zum Einsatz.

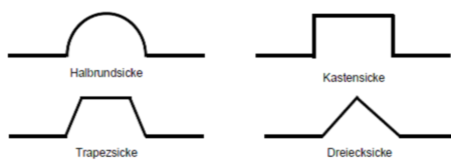


Abbildung 1: Sickenprofile [4]

[1] CAD steht für computer-aided design (engl.) also rechnerunterstütze Konstruktion. Dadurch können beispielsweise technische Zeichnungen an einem Bildschirm visualisiert werden.

[2] Ingenieurgesellschaft für intelligente Lösungen und Systeme mbH

In der Luft- und Raumfahrttechnik werden Sicken beispielsweise eingesetzt, um eine Gewichtsreduktion des benötigten Materials zu erreichen. Sie eignen sich daher hervorragend für die Konstruktion von Satellitenseitenwänden und reduzieren mit der dadurch erzeugten Schalenversteifung die Neigung zu Schwingungen.
Sicken ermöglichen unter anderem den Einsatz von dünnerem Blech, weshalb der Material- und Gewichtsaufwand, der für den Satelliten aufgebracht werden muss, minimiert wird. Die Verwendung von Sicken gewährleistet die erforderliche Stabilität, da auf die einzelnen Satellitenwände zeitweise enorme Kräfte einwirken. Schwierig gestaltet sich durch die geringe Blechdicke das Befestigen von benötigten Satellitenbauteilen an der dünnen Montageplatte. Die Befestigung der Satellitenbauteile erfolgt deshalb über speziell ausgestaltete Bohrlöcher. Diese Bohrlöcher dienen zusätzlich als Begrenzung, da die Sickenstege, die über die Bohrlöcher verlaufen, durch die dadurch entstandenen Verdickungen bzw. Erhöhungen eine optimale Grundlage zur Befestigung der Bauteile bieten.

3 DESIGN VON ENTWURFSSPRACHEN

Zur Erstellung einer graphenbasierten Entwurfssprache, die verschiedene technische Systeme beschreiben kann, wird auf einen Teil der UML[3] zurückgegriffen. Die verschiedenen Bauteile werden als Vokabeln abgebildet, die in einem UML-Klassendiagramm in Form von Klassen dargestellt werden. Anschließend werden diese Vokabeln in einem UML-Aktivitätsdiagramm durch die Definition von Regeln miteinander verknüpft. Durch diese Verknüpfung von Regeln entsteht ein maschinell im Compiler ausführbares Produktionssystem, das bei verschiedenen Anfangsbedingungen wiederholt ausgeführt werden kann um unterschiedliche Entwurfskonzepte zu erstellen. In der hier vorgestellten Arbeit wird das beschriebene Vorgehen verwendet, um dabei CAD-Modelle von verschiedenen versickten Montageplatten zu erstellen.

[3] UML steht für Unified Modeling Language. Dies ist eine graphische Modellierungssprache welche von der Object Management Group entwickelt wurde [5].

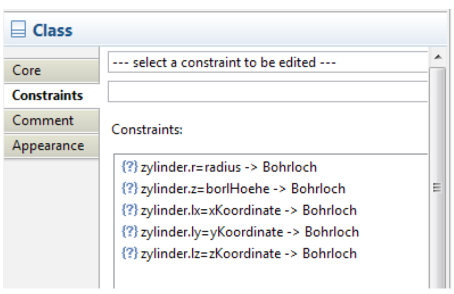


Abbildung 2: Einschränkungen der Bohrloch-Klasse

Die einzelnen Vokabeln der Entwurfssprache können mit unterschiedlichen Assoziationstypen verbunden werden. Dies wird in einem UML-Klassendiagramm spezifiziert. Des Weiteren besteht in einer graphenbasierte Entwurfs¬sprache die Möglichkeit, vorgegebene Gleichungen zur Berechnung bestimmter Klassenattribute einzusetzen. Dies ist in Abbildung 2 anhand einfacher Gleichungen zur Definition einer Instanz der Klasse „Bohrloch“ dargestellt, die im CAD-Programm über ein Zylinder-Primitiv visualisiert wird. Dabei entspricht der Radius „r“ des Zylinders dem Wert des Attributs „radius“ der Klasse „Bohrloch“. Hierbei sind Gleichungen beliebiger Komplexität möglich, wie sie durch nichtlineare algebraische Gleichungssysteme ausgedrückt werden können. Dieses Vorgehen ermöglicht eine flexible Berechnung der benötigten Werte.
Durch den Einsatz des UML-Klassendiagrammes werden die Vokabeln der Entwurfssprache realisiert. Diese Vokabeln werden in einem UML-Aktivitätsdiagramm über Regeln verbunden. Innerhalb dieser Regeln können vorhandene Instanzen verwendet werden. Es ist beispielsweise möglich, zwei vorhandene Instanzen über eine Assoziation miteinander zu verbinden. In der folgenden Abbildung 3 ist eine beispielhafte Regel dargestellt.

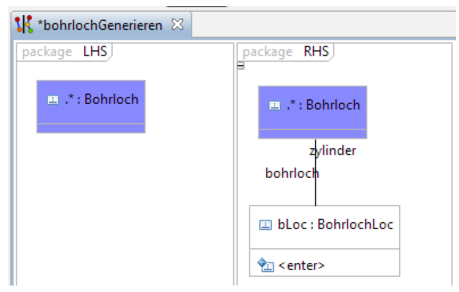


Abbildung 3: Aufbau einer Regel im UML-Aktivitätsdiagramm

Im linken Abschnitt (LHS) der Abbildung 3 können vorhandene Instanzen gesucht werden um sie für die aktuelle Regel zu benutzen. Im gezeigten Beispiel werden alle Instanzen der im Klassendiagramm angelegten Klasse "Bohrloch" gesucht. Anschließend wird diese im rechten Abschnitt (RHS) mit einer neuen Instanz verbunden. Dies geschieht in UML über eine Assoziation. Werden mehrere solcher Regeln miteinander verknüpft, entsteht daraus ein Aktivitätsdiagramm.

Die hier vorgestellte graphenbasierte Entwurfssprache in UML nutzt die Umgebung der Open-Source Programmierplattform Eclipse. Eclipse war anfänglich als integrierte Entwicklungsumgebung für die Programmiersprache Java entwickelt worden. Jede Technologie und jeder Quellcode, der von der Eclipse Foundation bereitgestellt wird kann dazu frei genutzt werden [6]. Bei der Entwicklung von Entwurfssprachen nutzt man diese Möglichkeit und generiert eigene Funktionalitäten, die mit Hilfe einer in Eclipse integrierten, offenen Plug-In-Struktur umgesetzt werden. Über den Einsatz dieser Plug-Ins können auch Anbindungen zu Drittsoftware erfolgen, wie beispielsweise CAD-Programme.

Bei der Entwicklung von Entwurfssprachen wurde verstärkt darauf geachtet, dass die Sprache bei jeder Modifikation eines Parameters auch alle Parameter neu berechnet, die mit dem Parameter gekoppelt sind. Dieses Vorgehen wird im folgenden Beispiel detaillierter beschrieben.

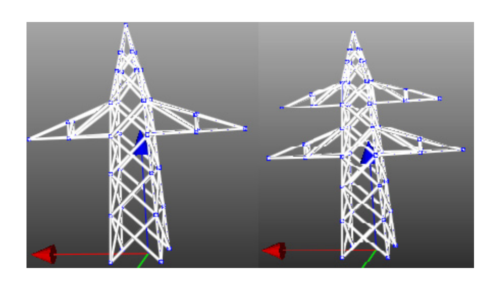


Abbildung 4: Beispiel eines Strommasten mit zwei bzw. vier Armen

In dem in Abbildung 4 dargestellten Schaubild eines Strommasten, kann beispielsweise die Anzahl der vorhandenen Arme anhand eines Parameters verändert werden. Das System modifiziert daraufhin selbstständig alle weiteren Parameter nach einem zuvor angelegten Muster. Dadurch werden beispielsweise Module, die als Platzhalter vorgesehen sind automatisch, hinzugefügt.

4 UMSETZUNG

Die Aufgabe der wissenschaftlichen Vertiefung bestand darin, einen Algorithmus zu entwickeln, der eine mögliche Platzierung von Sicken berechnet und visualisiert. Ein möglicher Einsatzort von Sicken findet sich in der Luft- und Raumfahrt. In dieser Arbeit wurde das Anbringen von Sicken an einer Satellitenwand unter der Berücksichtigung der Befestigungspositionen verschiedener Satelliten-bauteile realisiert.

Die Basis der umgesetzten Entwurfssprache bildet das speziell dafür erstellte Klassendiagramm (siehe Abbildung 6). Dort sind die benötigten Klassen mit den dafür erforderlichen Geometrieformen (z.B. Cuboid, Cylinder, Point) des Geometrie-Plug-Ins über Assoziationen miteinander verbunden. Für die Visualisierung der Satellitenwand wurde ein Quader gewählt, welcher die Darstellung der Montageplatte übernehmen soll. Die Bohrlöcher sind mittels Zylinderformen realisiert, da diese ihrem realen Aussehen am ähnlichsten sind. Als Sickenprofil wurde die in Abbildung 1 aufgezeigte Kastensicke verwendet, da sich diese mit den vorhandenen geometrischen Möglichkeiten gut realisieren lässt.

Für den Aufbau der Sicken müssen folgende Parameter manuell angegeben werden:

- Die Maße der Montageplatte: Breite / Tiefe / Höhe
- Höhe der Stege / Breite der Stege / Eckenradius der Stegübergänge welche gemeinsam die Sicke bilden
- Bohrlochhöhe / Bohrlochdurchmesser

Abhängig von der Plattengröße und den Positionen der Bohrlöcher werden die Größen der unterschiedlichen Sicken bestimmt. Zusätzlich werden die Koordinaten für die Positionierung der Sicken ermittelt.

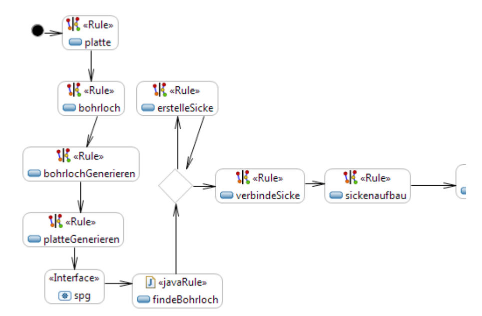


Abbildung 5: Ausschnitt des UML-Aktivitätsdiagrammes

Wie in Abbildung 7 aufgezeigt, befindet sich jedes Bohrloch auf einem Steg. Dadurch wird gewährleistet, dass nur das notwendigste an Material verbaut wird um Einsparungen zu erzielen. Der Ablauf des Projekts ist in einem UML-Aktivitätsdiagramm festgelegt, welches in Abbildung 5 teilweise abgebildet ist. Dort befinden sich auch die in Kapitel 3 beschriebenen Regeln, die durch Assoziationen miteinander verknüpft sind. Zusätzlich ist es möglich, den eigens entwickelten Java Code in das UML-Aktivitätsdiagramm mittels sogenannter Java-Regeln zu implementieren.

Das Aktivitätsdiagramm beginnt mit der Definiton von Axiomen (Anfangsbedingungen). Hierzu wurden Regeln (in Abbildung 5 durch „platte“ und „bohrloch“ gekennzeichnet) erstellt in denen die modifizierbaren Parameter festgehalten sind. Anschließend werden diese Parameter mit den dafür vorgesehenen geometrischen Primitiven verknüpft. Über ein internes Constraint Processing[4] (d.h. eine interne automatische Verarbeitung der Entwurfsgleichungen) werden die zu diesem Zeitpunkt erforderlichen und ggfs. noch unbestimmte Werte berechnet und die damit verbundenen Instanzen erzeugt. Daraufhin ist eine Java-Regel implementiert die auf diese Instanzen zugreift um die benötigten Parameter auslesen zu können. Der Ablauf dieser Java-Regel wird im folgenden Abschnitt genauer beschrieben.

Über verschiedene, in früheren Projekten vom ISD implementierte Java Methoden wird auf die einzelnen Werte der zuvor generierten Instanzen zugegriffen. Dadurch werden die Längen und Richtungen der Stege ermittelt aus welchen später die einzelnen Sicken zusammengebaut werden. Die Montageplatte wird in der Java-Regel von links nach rechts abgearbeitet. Das bedeutet, dass von der linken Randposition die Bohrlöcher in Richtung der rechten Randposition gesucht werden.

Sobald ein Bohrloch gefunden ist, überprüft der Algorithmus in welche Richtung der Steg verbaut werden kann. Hierzu werden die Distanzen zwischen dem davorliegenden sowie dem nachfolgenden Steg (rechts - links sowie auch oben - unten) miteinander verglichen. Die kürzeste Distanz zwischen den Stegen ist ausschlaggebend ob der Steg vertikal oder horizontal verbaut wird. Im Anschluss daran wird geprüft, ob weitere Bohrlöcher auf der Montageplatte vorhanden sind. Ist dies der Fall, wird der Vorgang so oft wiederholt bis alle Bohrlöcher abgearbeitet sind. Hat der Algorithmus ein Bohrloch ermittelt, das auf einem vorhandenen Steg liegt, wird dieses ignoriert um eine redundante Stegbildung zu vermeiden. Sobald die Java-Regel alle er-

[4] Als „constraint processing“ wird ein eigenes Gebiet in der Künstlichen Intelligenz (engl. artificial intelligence) bezeichnet, das sich mit der automatisierten Lösung von Entwurfsgleichungen (engl. design constraints) beschäftigt [7].

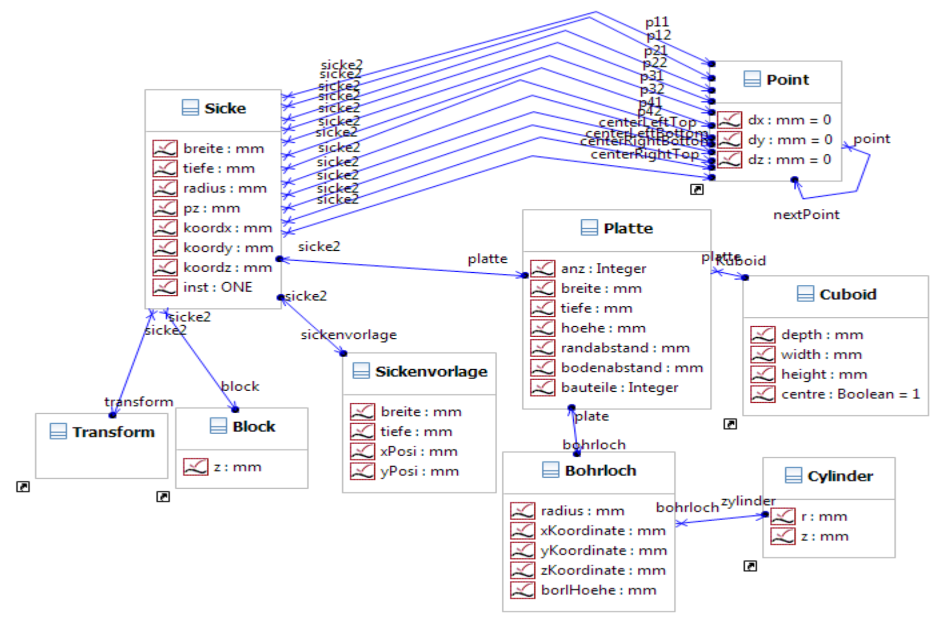


Abbildung 6: Klassendiagramm für die Sickenerstellung

forderlichen Daten (Längen und Richtungen der Stege sowie deren Koordinaten) ermittelt hat, werden diese an die UML Instanzen übermittelt um weiterverarbeitet werden zu können.
Im weiteren Verlauf des Aktivitätsdiagramm ist eine Regel definiert, die die Konstruktion der Sicken („erstelleSicke“) übernimmt. Damit die richtige Anzahl an Sicken garantiert ist, wird diese Regel für jede Sicke wiederholt durchlaufen. Dafür wurde im Aktivitätsdiagramm an der vorherigen Position ein „Decision Node“ (Entscheidungsknoten) eingefügt. Erst wenn alle benötigten Instanzen der Sicken erstellt wurden leitet der „Decision Node“ den Aktivitätsfluss zum nächsten Schritt weiter welcher den visuellen Sickenaufbau einleitet.
Dieser Aufbau findet wiederum in einer individuellen Regel statt. Es werden die ermittelten Stege (in Form von Linien als Längen und Breiten der Sicke) mit geometrischen Kreisbögen (als visuelle Variante der Ecken) verknüpft um eine einheitliche Sickenfläche zu generieren. Später wird diese Komposition in einen dreidimensionalen Körper transferiert. Hierzu werden die einzelnen Komponenten fest miteinander verknüpft und nach oben extrudiert.
Schlussendlich wird die endgültige Montageplatte generiert. Dazu müssen die einzelnen Sicken mit der Montageplatte verbunden werden. Anschließend werden die Bohrlöcher an den dafür vorgesehenen Positionen angeheftet, um das Schaubild in Abbildung 7 zeigen zu können.

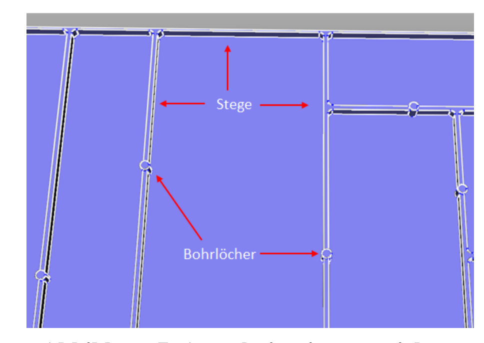


Abbildung 7: Ausschnitt einer versickten Montageplatte

5 FAZIT & AUSBLICK

Die vorgestellte Arbeit ermöglicht die Visualisierung von Sicken unterschiedlicher Größen auf einer Montageplatte ähnlich einer Satellitenwand. Dazu wurde eine Entwurfssprache entwickelt, die durch den Einsatz von UML und Java Code die prozedurale Generierung der Sicken ermöglicht. Dabei werden die einzelnen Bauteile als Vokabeln im UML-Klassendiagramm definiert. Zusätzlich wird der prozedurale Ablauf in einem UML-Aktivitätsdiagramm angelegt. Hierfür werden spezielle Regeln erstellt, durch die die einzelnen Arbeitsschritte abgearbeitet werden. Es sind dafür nur einige Parameter über die Montageplatte und ihre Bohrlöcher nötig, welche als Startparameter angegeben werden müssen. Die entwickelte Entwurfssprache modelliert daraus eigenständig die benötigten Sicken. Das Ergebnis stellt ein dreidimensionales Modell dar, das in einem CAD-ähnlichen Plug-In als Geometriemodell (vergleichbar mit Modellen in Catia oder OpenCascade) ausgegeben wird. Dieses kann direkt in der Entwicklungsumgebung visualisiert werden.

In der aktuellen Ausführung sind bisher nur rechtwinklige Sicken in Form von Quadraten oder Rechtecken möglich. Eine zukünftige Entwicklung für flexiblere Sickenformen wäre sinnvoll. Dabei sollten beispielsweise diagonale Stege berücksichtigt werden. Ebenfalls ist es denkbar weitere Sickenprofile (siehe Abbildung 1) zu realisieren, um ein noch breiteres Spektrum für die mechanische Herstellung zu bieten.

6 LITERATURVERZEICHNIS

[1] Alber, R. and Rudolph, S. 2003.“43“ – a generic approach for engineering design grammars. Proc. of the AAAI Spring Symposium – Computational Synthesis, Stanford.

[2] Kröplin, B. and Rudolph, S. 2005. Entwurfsgrammatiken – Ein Paradigmenwechsel? Der Prüfingenieur. S. 34 – 43.

[3] Widmann, M. 1984. Herstellung und Versteifungswirkung von geschlossenen Halbrundsicken. Springer Verlag, Berlin.

[4] Emmrich, D. 2005. Entwicklung einer FEM-basierten Methode zur Gestaltung von Sicken für biegebeanspruchte Leitstützstrukturen im Konstruktionsprozess. Dissertation an der Universität Karlsruhe.

[5] Object Management Group
OMG Unified Modeling Language (OMG UML), Infrastructure. August 2011.
Abrufbar unter: http://www.omg.org/spec/UML/2.4.1/Infrastructure/PDF/. Letzter Zugriff 01.03.2012.

[6] The Eclipse Foundation
http://www.eclipse.org/org/.
Letzter Zugriff: 24.02.2012.

[7] Rudolph, S. and Bölling, M. 2004. Constraint-based conceptual design and automated sensitivity analysis for airship concept studies. Aerospace Science and Technology 8.

Humanoide Serviceroboter

Hannah-Elena Seeger
Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Prof. Dr. rer. nat. Gabriela Tullius
Hannah-Elena.Seeger@Student.Reutlingen-University.de

ABSTRACT

In diesem Fachartikel wird das Gebiet der humanoiden Serviceroboter und deren Einsatz im Alltag vorgestellt. Es werden die Schwierigkeiten bei der Gestaltung und Umsetzung eines alltagstauglichen Roboters, der mit Menschen interagieren, kommunizieren und unterstützende Tätigkeiten ausführen kann, erläutert. Desweiteren werden drei humanoide Roboter vorgestellt und im Hinblick auf ihre Servicefähigkeiten analysiert.

1 EINLEITUNG

Serviceroboter: entwickelt um den Menschen bei den alltäglichen Aufgaben zur Hand zur gehen oder die Aufgaben direkt automatisiert auszuführen. Bereits 1994 definierte das Fraunhofer IPA einen Serviceroboter als:
„eine frei programmierbare Bewegungseinrichtung, die teil- oder vollautomatisch Dienstleistungen verrichtet. Dienstleistungen sind dabei Tätigkeiten, die nicht der direkten industriellen Erzeugung von Sachgütern, sondern der Verrichtung von Leistungen für Menschen und Einrichtungen dienen.“[6]
Als diese Definition formuliert wurde, gab es wenige Serviceroboter. Heute gibt es eine Vielzahl an kommerziell erhältlichen Serviceroboter. Beispielsweise Staubsaugroboter, Mähroboter, Fensterputzroboter, Reinigungsroboter, Wachroboter und Pflegeroboter. Um Pflege- und Haushaltsroboter effektiver zu gestalten, wird im Bereich der multisensoriellen Perzeption, der Kognition, der Mensch-Maschine-Interaktion und Kooperation sowie der Künstlichen Intelligenz geforscht. Dabei werden vor allem die Gesetze und Abläufe der Biologie als Vorbilder für Roboter angesehen. Um das gesamte Ausmaß der biologischen Komplexität technisch umzusetzen, versuchen die Forscher als höchstes Ziel einen Roboter nach ihrem Ebenbild zu entwickeln. Diese humanoiden Roboter sollen als Helfer und Freund des Menschen interagieren und kooperieren können. Wie weit die Entwicklung voran gegangen ist, zeigen die in den nächsten Kapiteln vorgestellten humanoiden Roboter. Zunächst stellt sich jedoch die Frage: wie menschlich darf ein Roboter sein und wie menschlich muss er sein?

2 UNCANNY VALLEY [5]

Forscher, auf dem Gebiet der Robotik und der Wahrnehmungspsychologie haben durch Untersuchungen herausgefunden, dass es für Menschen leichter ist, mit einem humanoiden Roboter zu interagieren, wenn er ein menschliches Aussehen besitzt. Dies liegt an der Fähigkeit des Menschen, zusätzlich zu der verbalen Kommunikation auch die nonverbale Kommunikation wahrzunehmen, wie zum Beispiel die Augenbrauenbewegung.
Masahiro Mori entdeckte bei seinen Untersuchungen das „Uncanny valley“-Problem der humanoiden Roboter. Abbildung 1 verdeutlicht das Problem: wenn ein Roboter menschenähnlicher wird, steigt die Vertrautheit zwischen Roboter und Mensch bis zu einem bestimmten Punkt an. Ab diesem Punkt wird die Wahrneh-

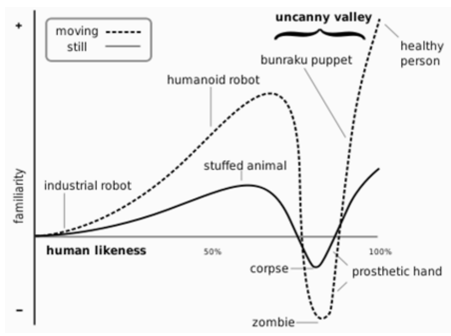


Abbildung 1: Uncanny valley Problem [5]

mung der Menschenähnlichkeit und gleichzeitig der Technik dahinter als negativ angesehen und damit dem Menschen unheimlich. Wird der Roboter noch menschenähnlicher, wird er wieder als positiv und vertraut angesehen.

3 HUMANOIDE SERVICE-ROBOTER

In den folgenden Unterkapiteln werden drei humanoide Roboter im Bezug auf ihre Servicefähigkeit analysiert.

3.1 NAO [1][2]

Die Firma Aldebaran hat einen 57 Zentimeter großen und 4,3 Kilogramm schweren humanoiden Roboter entwickelt. Der Roboter NAO wurde mit 25 Freiheitsgraden ausgestattet, wodurch eine hohe Beweglichkeit des Roboters erreicht wurde. Um das Umfeld und die Umgebung wahrzunehmen, verfügt der NAO über zwei Kameras, 4 Mikrofone, 2 Infrarotempfänger und –sender, 9 taktile Sensoren und 8 Drucksensoren. Mithilfe zweier Lautsprecher, einem Stimmensynthesizer und mehreren LED-Lampen ist der NAO in der Lage mit den Menschen in seinem Umfeld zu kommunizieren. Durch seine Gestaltung und seine Größe besitzt er gegenüber den Menschen in seiner Umgebung einen gewissen Niedlichkeitsfaktor, welcher die Kommunikation und Interaktion mit dem Roboter erleichtert (siehe Abbildung 2).



Abbildung 2: Design des humanoiden Roboter NAO [1][2]

Für den Einsatz als Kommunikationspartner ist der NAO dank der Spracherkennung geeignet. Die Akkuleistung des Roboters ist jedoch nicht ausreichend für einen Einsatz als Serviceroboter. Unter Dauerbelastung ist der Akku bereits nach 15 Minuten erschöpft. Bei leichter Belastung kann die Leistung des Akkus bis zu 45 Minuten ausreichen. Ein weiteres Problem für den Einsatz als Serviceroboter stellt die Überhitzung des Roboterkopfes und der Gelenke dar. Bewegt sich der NAO viel oder muss zum Beispiel seinem Gegenüber über längere Zeit zuhören, erhitzt sich der Roboterkopf, in dem sich der Prozessor des Roboters befindet und die Gelenkmotoren. Das größte Problem für den Einsatz als Serviceroboter stellen die vorgegebenen Programmabläufe dar. Der Roboter wird mithilfe einer eigens entwickelten Software programmiert. Ist das Programm fertiggestellt, wird dieses an den Roboter übertragen. Der NAO führt das Programm in der vorgegebenen Reihenfolge aus und beendet es anschließend. Eine dynamische Änderung des Programmablaufs des NAO-Roboters, um eine Anpassung an Umweltänderungen zu erreichen, ist nicht vorgesehen. Der Roboter kann Gesichter erkennen und darauf reagieren. Die Kopfkamera kann Erwachsene, aufgrund der kleinen Größe des NAOs und des damit geringen Aufnahmewinkels, nicht aus nächster Nähe erfassen. 2011 wurde ein neuer NAO-Roboters vorgestellt. Verbesserungen im Rahmen der nächsten Generation wurden beispielsweise im Bereich der Prozessorleistung und der Kamera vorgenommen.

3.2 KASPAR [4]

Im Rahmen der Adaptive Systems Research Group an der Universität von Hertforshire wird seit 2006 an einem humanoiden Roboter mit dem Namen KASPAR geforscht. Dieser Roboter wurde für soziale Interaktionen entwickelt und mit der Fähigkeit versehen, minimale Gesichtsausdrücke darzustellen. Der Roboter wird eingesetzt, um die Beziehungen zwischen einer Interaktion von Menschen und Maschine zu studieren. Dazu werden Gesten, Mimik, Synchronisation und Imitationen eingesetzt. KASPAR wird für Verhaltens- und Entwicklungsforschungen eingesetzt vor allem



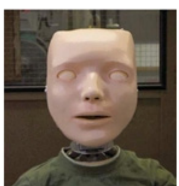


Abbildung 3: Mimik des KASPAR [4]



Abbildung 4: Design des KASPAR [4]

im Bereich des Autismus. Die Adaptive Systems Research Group möchte herausfinden, ob ein humanoider Roboter Kindern, mit Entwicklungsverzögerungen und fehlenden sozialen Fähigkeiten, als eine Art Therapeut dienen kann. Aus diesem Grund wurde der Roboter mit einem kindlichen Aussehen entwickelt. Das Aussehen des Roboters sollte nicht absolut der Realität entsprechen, sondern realistisch genug für eine Vielzahl an Interaktionen sein. KASPAR besitzt im Kopf- und Nackenbereich acht Freiheitsgrade und sechs weitere Freiheitsgrade in den Armen und Händen. Das Gesicht und die Hände des Roboters sind mit einer Silikonhülle versehen, um das Interagieren für Kinder zu erleichtern, indem man ihnen die Scheu vor der Technik nimmt. KASPAR besitzt integrierte Kameras in den Augen. Die Augen besitzen zwei Freiheitsgrade. Desweiteren ist der Roboter in der Lage den Mund zu öffnen und zu lächeln. KASPAR ist nicht in der Lage zu gehen, er sitzt auf einem Tisch in einer Position, die für den Benutzer gemütlich, verspielt und entspannt aussehen soll (siehe Abbildung 4). Die Sitzposition wurde damit an die Sitzweise eines spielenden Kindes angelehnt.

Als Spielkamerad zum Fangen spielen, rennen oder verstecken kann der Roboter damit nicht eingesetzt werden. Durch den Aufbau der Arme und Hände ist der Roboter jedoch in der Lage zu gestikulieren und mit Objekten zu interagieren. Der Roboter wurde mit Augenlidern versehen um eine humanoide Mimik erzeugen zu können (siehe Abbildung 3).

Um während einer Studie den Teilnehmer nicht durch Computer zu beeinträchtigen, kann der Roboter mit einer Fernbedienung gesteuert werden. Diese Fernbedienung ist eine kabellose Tastatur mit 20 Tasten, mit deren Hilfe verschiedene Verhaltensabläufe aufgerufen werden können. Die Verhaltensreihenfolge ist damit bei KASPAR vorbestimmt und lässt wenig Spielraum für eine Änderung der Abläufe.

3.3 RH-1 [3]

Im Systems Engineering and Automation Department des Robotic Lab der Carlos III Universität in Madrid wird seit 2005 an der Entwicklung eines humanoiden Roboters mit dem Namen Rh-1 gearbeitet. Der Roboter besitzt die Größe und das Gewicht einer 1,20m-großen Person und die Fähigkeit auf verschiedenen Terrains zu gehen. Desweiteren soll er in der Lage sein Treppen steigen zu können.

Der humanoide Roboter kann mit leichten Objekten (bis zu 750g) interagieren. Mithilfe der im Kopf integrierten Sensoren, erkennt der Rh-1 seine Position und die Richtung in die er sich bewegen soll. Durch den Aufbau des Roboters mit 21 Freiheitsgraden und Aluminium wird ein menschenähnliches Gehen des Roboters erreicht. Das Ziel der Forschung am Rh-1, ist die Entwicklung eines Roboters mit einem möglichst geringen Gewicht und einer maximalen Beweglichkeit.

Die Forschungen und Entwicklungen der Mitarbeiter des Robotics Lab an der Universität in Madrid fokussieren sich auf das menschen-

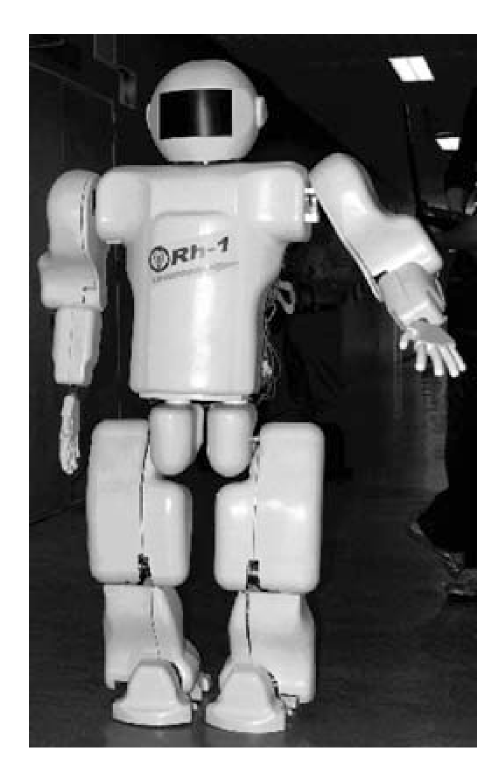


Abbildung 5: Rh-1, der laufende humanoide Roboter [3]

ähnliche Laufen des Rh-1. Für den Einsatz des Rh-1 als humanoider Serviceroboter ist das menschenähnliche Laufen ein wichtiger Punkt, jedoch fehlen die Interaktions- und Kooperationsmöglichkeiten. So müsste der Roboter um eine Spracherkennung oder Gestik und Mimik erweitert werden.

4 FAZIT

Die Analyse verschiedener humanoider Serviceroboter hat gezeigt, dass einzelne Anforderungen an einen humanoiden Roboter als Helfer und Freund im Alltag erfüllt wurden .Beispielsweise die Fortbewegung oder die Gesichtserkennung. In den Bereichen der Objektinteraktion, der Gestik und Mimik und der künstlichen Intelligenz muss noch weiter geforscht und entwickelt werden. So benötigt die Forschung weitere Jahre der Entwicklung, um einen frei interagierenden und kooperierenden Serviceroboter in menschlicher Form zu entwickeln, der alle Bereiche abdeckt. Die entwickelten Ansätze, die in diesem Artikel vorgestellt wurden sind vielversprechend, doch die Lösung der zahlreichen Probleme bei der Entwicklung wird die Robotikforschung weiterhin beschäftigen. Die wichtigsten Probleme die gelöst werden müssen, sind die kurze Batterielaufzeiten, die menschenähnliche Fortbewegung, die künstliche Intelligenz und die Spracherkennung.

5 LITERATURVERZEICHNIS

[1] Aldebaran Robotics. 2012. Firmenwebseite. http://www.aldebaran-robotics.com/en/Home/welcome.html?language=en-GB

[2] Aldebaran Robotics. 2012. NAO Software Documentation 1.12. http://www.aldebaran-robotics.com/documentation/index.html

[3] Arbulú, M., Kaynov, D., Cabas, L. and Balaguer, C. 2009. The Rh-1 full-size humanoid robot: Design, walking pattern generation and control. Applied Bionics & Biomechanics. Sep 2009, Vol.6 Issue ¾, p301-344, 44p, 13 Black and White Photographs, 35 Diagrams, 2 Charts, 13 Graphs.

[4] Dauthenhahn, K., Nehaniv, C., Walters, M., Robins, B., Kose-Bagci, H., Mirza, N., and Blow, M. 2009. KASPAR - a minimally expressive humanoid robot for human-robot interaction research. Applied Bionics & Biomechanics. Sep 2009, Vol. 6 Issue 3/4, p369-397, 29p, 27 Color Photographs, 3 Diagrams, 1 Chart, 1 Graph.

[5] Decker, M.. 2007. Ein Abbild des Menschen: Humanoide Roboter. Information und Menschenbild - Ethics of Science and Technology Assesment. 2007, Bd. Volume 37, DOI: 10.1007/978-3-642-04742-8_3.

[6] Schraft, R.D., Hägele, M., Wegener, K. 2004. Service Roboter Visionen. Carl Hanser Verlag München Wien. ISBN 3-446-22840-3

Intuitive Interaktionsmöglichkeiten in virtueller Realität

Anja Rothbart
VRLab / Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Anja.Rothbart@Student.Reutlingen-University.de

ABSTRACT

Innovative Lösungen zum Thema Interaktion zwischen Mensch und Maschine in einer virtuellen Umgebung spielen in Wirtschaft und Wissenschaft eine immer größer werdende Rolle. Aus diesem Grund beschäftigen sich Masterstudenten der Hochschule Reutlingen im Studiengang Medien- und Kommunikationsinformatik mit ausgewählten Forschungsthemen der virtuellen Realität. Seit sechs Jahren bietet das *Virtual Reality Laboratory* (VRLab) Schwerpunkte in den Bereichen Interaktion, Motion Tracking zur Steuerung virtueller Welten sowie stereoskopischer Projektion von medialen und dreidimensionalen Inhalten [1]. Eine intuitive Bedienung durch den Nutzer ist bei der Entwicklung der Interaktionsverfahren besonders zu beachten. Im Folgenden wird das VRLab und drei der aktuellen Projekte vorgestellt, die sich mit *natural user interfaces* (NUI) befassen.

1 VIRTUELLE REALITÄT

Virtuelle Realität beschreibt „[…] eine mittels Computer simulierte Wirklichkeit oder künstliche Welt, in die Personen mithilfe technischer Geräte sowie umfangreicher Software versetzt und interaktiv eingebunden werden." [2] Ziel ist es, eine simulierte Umgebung zu erschaffen, in der Benutzer interagieren und Anwendungen steuern können [3].
Das VRLab beschäftigt sich dabei hauptsächlich mit natürlichen Interaktionsmöglichkeiten. Die Entwicklung natürlicher User-Interfaces vereinfacht die Steuerung zwischen Mensch und Maschine über einen intuitiven Weg der Eingabemöglichkeit [4]. Benutzer sollen so schnell wie möglich verstehen können, wie sie mit der virtuellen Umgebung interagieren können.



Abbildung 1: Interaktion mit dem Multi-Touch Cube

Das VRLab legt den Schwerpunkt dabei auf Multi-Touch-Geräte und Gestensteuerung mittels der Microsoft Kinect. Von Bedeutung ist hierbei die Entwicklung und Integrierung von Low-Cost-Ansätzen [1].

2 MULTI-TOUCH SYSTEME

Bekannt geworden sind Multi-Touch-Systeme hauptsächlich durch die immer größer werdende Aufmerksamkeit um Smartphones und Tablet-PCs. Die Multi-Touch-Technologie dieser Geräte wird meist über das kapazitive[1] Verfahren realisiert. Da im VRLab jedoch vor allem Low-Cost-Lösungen umgesetzt werden sollen, wurde die Multi-Touch-Funktionalität über das optische Verfahren entwickelt [5].
Die Projektionsfläche des Gerätes wird bei diesem Ansatz mit im Innenraum platzierten Infrarot (IR) LEDs möglichst gleichmäßig ausgeleuchtet. Wird die Projektionsfläche vom Benutzer berührt, kann die eingebaute Kamera mit Hilfe eines integrierten IR-Bandpassfilters das von den Fingern diffus reflektierte Licht aufzeichnen. Dieser Ansatz wird als *Rear Dif-*

[1] vgl. S. Cashman Series 2011 Discovering Computers – Living in a Digital World. S.430



Abbildung 2: Multi-Touch Tisch

fused Illumination Technik bezeichnet. Mit Hilfe der Software *Community Core Vision*[2] (CCV) können die Koordinaten der Berührungspunkte aus dem Kamerabild berechnet und in eine Berührung der Projektionsfläche umgewandelt werden. Diese Umwandlung erfolgt unter der Benutzung des TUIO[3] Protokolls, das die berechneten Daten an die Multi-Touch-Anwendung weiterleitet [1].

Im VRLab wird aktuell an zwei Multi-Touch-Systemen gearbeitet, die im Folgenden vorgestellt werden.

2.1 MULTI-TOUCH-TISCH

Eines der Multi-Touch-Systeme verfügt über eine horizontal ausgerichtete Bedienfläche und wurde von den Studenten des VRLabs als Multi-Touch-Tisch selbst konzipiert. Durch einen im Boden angebrachten Spiegel kann das Bild eines LED-Projektors auf der Projektionsfläche dargestellt werden. Die Bedienoberfläche des Tisches besteht aus einer halbtransparenten Acrylglasscheibe. Die IR-LEDs mit einer Wellenlänge von 940 nm wurden in Form von 500 mm langen Streifen im Innenraum des Tisches platziert. Der Innenraum wurde zusätzlich mit Molton ausgekleidet, um störende Reflektionen im IR Spektrum zu verhindern. Die Trackingpunkte werden mit Hilfe einer Playstation 3 Eye Cam aufgezeichnet.

Durch eine von den Studenten selbst errichtete Verkleidung (siehe Abbildung 2) konnte eine stabile und zugleich mobile Benutzung des Tisches realisiert werden. Es ist bereits möglich native Multi-Touch-Anwendungen oder das Betriebssystem Windows 7 durch Berührungen zu steuern [1].

2.2 MULTI-TOUCH-CUBE

Das zweite Multi-Touch-System ist ein Rückprojektionssystem mit vertikaler Bildschirmfläche. Auf Grund seiner würfelartigen Form trägt dieses System den Namen Cube. Bereitgestellt wurde das ursprünglich als Ausgabegerät konzipierte Großbildsystem von der eyevis GmbH[4]. Durch die Verwendung von Low-Cost Komponenten konnte daraus ein Multi-Touch-Gerät entwickelt werden.

Die Multi-Touch-Funktionalität beim Cube ist dem Aufbau des Multi-Touch-Tisches sehr ähnlich. Lediglich die Anordnung der einzelnen Low-Cost-Komponenten ist unterschiedlich. Statt im Boden wurde der Spiegel schräg zur Projektionsfläche angebracht. Auf Grund der Gehäusebeschaffenheit muss die Kamera an derselben Position wie der DLP-Projektor platziert werden. Da die Bedienoberfläche des Cubes sehr groß ist (1,1 m x 1,38 m) wurde alternativ zur Playstation 3 Eye Cam eine Firefly MV von Point Grey verwendet, die mit einem Weitwinkelobjektiv ausgestattet ist. Dadurch ist es möglich die Berührungen an den Rändern der Projektionsfläche besser zu erfassen. Des Weiteren kann die Kamera über ein Kugelkopfgelenk in ihrer Position und Orientierung optional bewegt werden [1].



Abbildung 3: Aufbau (innen) Multi-Touch Cube

[2] vgl. http://ccv.nuigroup.com

[3] vgl. http://www.tuio.org

[4] vgl. http://www.eyevis.de

2.3 NUTZERTEST

Im Wintersemester 2011/12 wurde von den Studenten ein Nutzertest zur Bedienung des Multi-Touch Cubes entwickelt und durchgeführt. Grund dieses Nutzertests ist der große Unterschied zu bereits existierenden herkömmlichen Multi-Touch-Geräten. Sowohl die große Projektionsfläche als auch die vertikale Ausrichtung der Bedienoberfläche sind dabei ungewöhnlich. Aspekte wie die Verdeckung der Inhalte durch den Nutzer, die optimale Positionierung von aktiven bzw. nicht aktiven Elementen oder die Ermüdung des Nutzers müssen vor der Entwicklung einer Multi-Touch-Anwendung evaluiert werden. Um eine möglichst benutzerfreundliche Anwendung entwickeln zu können, müssen diesbezüglich Gestaltungsrichtlinien herausgearbeitet werden. Die Anwendung für den Nutzertest wurde mit Hilfe von MT4J entwickelt. „MT4J ist ein Java-Framework zur Entwicklung von 2- und 3D MT Anwendungen, das unter der GNU General Public License (GPL) veröffentlicht wurde und auf OpenGL sowie Processing basiert.“ [1]

2.3.1 TESTAUFBAU

An dem Test nahmen insgesamt 18 Probanden teil, wobei 15 Personen zwischen 18 und 30 Jahren alt waren und lediglich drei zwischen 31 und 70. Zwölf der Probanden waren männlich, sechs weiblich. Die Testpersonen waren hauptsächlich Rechtshänder, lediglich vier Teilnehmer gaben eine Linkshändigkeit an.
Der Test bestand aus insgesamt drei Aufgaben. Im Wesentlichen mussten die Probanden eine Menüleiste am Rand des Cubes finden und bedienen. Auf diese Weise konnte ermittelt werden, auf welcher Seite (links/rechts/oben/unten) eine Menüleiste für den Benutzer in einer angenehmen Position angebracht werden sollte. Ebenfalls wurde getestet, ob die Menüführung über eine Wischfunktion als solche erkannt wird. Des Weiteren wurde ein Fitts Law[5] Test durchgeführt, um herauszufinden an welcher Stelle die Bedienelemente idealerweise platziert werden sollen [6].

2.3.2 ERGEBNISSE

Bei der Auswertung konnten folgende Erkenntnisse gewonnen werden:
Die Testpersonen hatten einige Probleme mit der Größe der Projektionsfläche. Die Bedienoberfläche ist zu groß, um vom Nutzer vollständig eingesehen werden zu können, da der Nutzer für die Interaktion nah am Bildschirm steht. So sind die am Rand platzierten Menüleisten nicht gleich zu erkennen und fordern eine genaue Sicht des Benutzers.
Ebenso empfanden mehr als die Hälfte der Probanden die Menüleiste im unteren Bereich des Cubes als unangenehm. Die Menüleiste oben wurde von den Testpersonen als angenehm bewertet, da die Wischfunktion an dieser Position intuitiver und den Benutzern bereits von aktuellen Touch-Geräten bekannt war. Jedoch müssen hierbei die unterschiedlichen Körpergrößen der Probanden beachtet werden. Kleinere Teilnehmer mussten sich strecken, um die Bedienelemente im oberen Bereich erreichen zu können. Allgemein wurde die Wischfunktion aber als positiv erachtet, sollte aber deutlicher dargestellt werden. Ebenso wird bei der Farbwahl der Menüleiste eine kräftige Farbe erwartet, die sich vom Hintergrund abhebt.
Durch die Größe der Interaktionsfläche mussten die Testpersonen weite Wege mit der Hand zurücklegen oder gegebenenfalls umgreifen. Interessant dabei ist aber, dass fast alle Teilnehmer den Cube mit einer Hand bedienten. Aus diesem Grund fanden rechtshändige (linkshändige) Probanden das Interagieren mit der linken (rechten) Hälfte des Cubes als unangenehm [6]. Die Ergebnisse des Nutzertests müssen bei der Implementierung einer Anwendung in jedem Fall beachtet werden.

3 KINECT

Als eine weitere natürliche Interaktionsmöglichkeit besteht neben der Multi-Touch-Technologie die freihändige Interaktionsform, mediale und dreidimensionale Inhalte über Gestensteuerung zu bedienen. Dem VRLab war es auch in diesem Bereich möglich mittels der Microsoft Kinect (siehe Abbildung 4) eine kostengünstige Lösung der Gestensteuerung zu realisieren.

[5] vgl. http://www.interaction-design.org/encyclopedia/fitts_law.html



Abbildung 4: Microsoft Kinect

Mit Hilfe eines Infrarot-Tiefensensors und einer RGB-Kamera können Körperbewegungen des Benutzers erfasst und schließlich über Bildverarbeitungsverfahren selektiert werden [7].
Um ein natürliches User-Interface innovativ steuern zu können, werden komplexere Gesten benötigt, als bisher vorgesehen. Deshalb wird im VRLab ein Ansatz entwickelt, „[…] Gesten unter Verwendung eines künstlichen neuronalen Netzes zu erkennen.“ [1] Das Feedforward-Netzwerk [8], das mit Hilfe des Backpropagation-Algorithmus [9] trainiert wird, diente für das Projekt der intuitiven Interaktionsentwicklung als Basis [1].
Für die Umsetzung wurde das .NET Encog Framework 3.0 [10] verwendet. Mit Hilfe der entwickelten Software können sowohl Trainings- als auch Testdaten aufgenommen und anschließend ausgewertet werden. Diese Daten liegen zunächst in Form einer Liste von 3D-Koordinaten vor. Dadurch kann die Distanz zweier benachbarter 3D-Punkte leicht berechnet werden, da die Koordinaten auf den Achsen x, y und z exakt bestimmt werden können. Die gemessenen Punkte werden anschließend in einem Vektor gespeichert, der als Eingabe für das neuronale Netz dient. „Die Ausgabe sieht ein Neuron für jede Geste vor […] [1]:

- 0 für: die Geste wurde nicht erkannt
- 1 für: die Geste wurde erkannt

In einem ersten Test wurden, basierend auf den Buchstaben A bis Z, 26 unterschiedliche komplexe und zusammenhängende Gesten verwendet, wobei eine überwiegend gute Erkennungsrate erreicht werden konnte. Die Durchführung einer repräsentativen Evaluation mit mehreren Probanden steht bisher noch aus [1].

4 AUSBLICK

Im nächsten Schritt gilt es, die bisher bestehenden natürlichen User-Interfaces zu evaluieren und eine darauf basierende Anwendung zu entwickeln. Der Nutzertest für den Multi-Touch-Cube hat bereits gezeigt, dass eine sorgfältige Betrachtung der Usability nicht außer Acht gelassen werden darf, um eine intuitive Anwendung entwickeln zu können. Aktuell befindet sich bereits eine Multi-Touch funktionale Anwendung in Planung, die sich mit dem Thema virtueller Realität im Bereich der Medizintechnik befasst.

5 LITERATURVERZEICHNIS

[1] Jannasch, M., Ludl, D., and Vöhringer C. 2011 Low-Cost NUI. Fachkongress der Gesellschaft für Informatik e.V.

[2] Brockhaus (ed.) 1997 Brockhaus – Die Enzyklopädie. 20., überarbeitete Auflage. F. A. Brockhaus GmbH Leipzig-Mannheim

[3] Brill, M. 2009 Informatik im Fokus - Virtuelle Realität. Springer Verlag

[4] Villaroman, N., Rowe, D., and Swan, B. 2011 Teaching Natural User Interaction Using OpenNI and the Microsoft Kinect Sensor. West-Point, New York, USA

[5] NUI Group Authors 2009 Multitouch Technologies, NUI Group, Community Release

[6] Jannasch, M., Vöhringer, C. 2012 Nutzertest Multitouch Cube Dokumentation (unveröffentlicht)

[7] Engineering and Technology Magazine 2011 The Teardown: The Kinect for Xbox 360. Engineering & Technology, Vol.6, Nr.3; S.94-95

[8] Haykin, S. 1999 Neural networks: a comprehensive foundation. Prentice Hall, Upper Saddle River, N.J., 2. Auflage

[9] Werbos, P. 1990 Backpropagation through time: what it does and how to do it. In: Proc. of the IEEE, Vol.78, Nr.10, 1990; S.1550-1560.

[10] Haeton Research, Inc.Encog Java and DotNet Neural Network Framework. http://www.heatonresearch.com/encog

Kollaborationskonzepte für wmc²

André Antakli
wmc² / Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Andre.Antakli@Student.Reutlingen-University.de

Marie Friedrich
wmc² / Hochschule Reutlingen
Medien- und Kommunikationsinformatik
Marie.Friedrich@Student.Reutlingen-University.de

ABSTRACT

Kollaborative Software unterstützt eine effektive und effiziente Zusammenarbeit. Dieser Artikel präsentiert den aktuellen Stand der Entwicklungen für das Masterprojekt wmc². Für die Entwicklung einer kollaborativen Software wird an einem modular aufgebauten Portalsystem gearbeitet, welches Clients auf unterschiedlichsten Geräten integriert. Aktuell werden zwei Kollaborationskonzepte entwickelt. Zum einen wird ein passendes Interface mit Einbezug von veränderten Nutzerverhalten durch mobile Endgeräte erstellt. Zum anderen werden Kollaborationsformen, wie zum Beispiel die Darstellung und Bearbeitung von Dateien, in einer virtuellen 3D-Welt abgebildet.

1 EINLEITUNG

wmc² steht für web, mobile, content und collaboration. Es ist ein Masterprojekt der Hochschule Reutlingen, das die Studierenden anregen soll, sich auf wissenschaftlichem Niveau Gedanken zum Thema „Kollaboration in der Zukunft" zu machen. Ziel des Projekts wmc² ist die Zusammenarbeit von Menschen mit Hilfe von elektronischen Geräten orts- und geräteunabhängig zu gestalten. Hinsichtlich der steigenden Mobilität soll jeder von jedem Ort aus die Möglichkeit haben mit anderen in Kontakt zu treten oder Inhalte auszutauschen. Aufgrund der zahlreichen Plattformen und mobilen Endgeräten will wmc² ein Kollaborationsportal schaffen, das von jeder Plattform und von jedem Gerät aus betreten werden kann.

2 DAS PROJEKT WMC²

2.1 ANFORDERUNGEN

Die Anforderungen an das Portal wurden durch eine Umfrage an der Hochschule Reutlingen im Frühjahr 2011 ermittelt. Durch die Befragung von Studenten und Mitarbeitern kam diese Umfrage zu folgenden Ergebnissen: Die Mehrheit der Befragten sahen als wichtigsten Bestandteil kollaborativer Software Funktionen für asynchrone Kommunikation und die Nutzung des Portals über mobile Endgeräte. [1]

2.2 ARCHITEKTUR

Im Rahmen von wmc² soll ein Portal für orts- und geräteunabhängige Zusammenarbeit geschaffen werden. Hierfür sieht das grundlegende Konzept von wmc² einen modularen Aufbau vor, der den Zugriff auf vorhandene Kollaborationswerkzeuge erlaubt. Der modulare Aufbau wird mit Hilfe von Webservices als Transaktionsdienste realisiert, die über eine

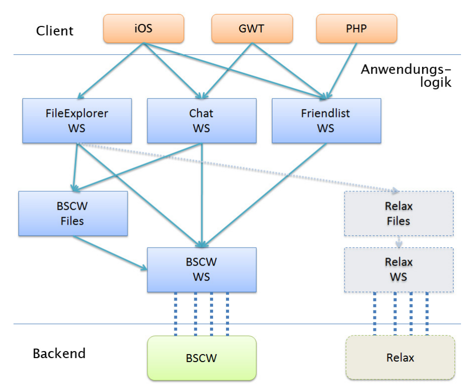


Abbildung 1: Experimentelle Architektur von wmc² in der Hochschulumgebung



XML-Schnittstelle und damit unabhängig von Programmiersprachen kommunizieren können. Der modulare Aufbau hat den Vorteil zukünftigen Anforderungen gewachsen zu sein. [1]
Exemplarisch wird für die Umsetzung auf die an der Hochschule Reutlingen vorhandenen Kollaborationswerkzeuge wie beispielsweise BSCW und Moodle (Relax) zurückgegriffen. Innerhalb des Projekts wird Software auf der Client- und Anwendungslogikebene entwickelt. Wie in Abbildung 1 zu sehen, greifen verschiedene Clients, Web-Clients oder mobile Clients, auf die Webservices zu. Die Webservices wiederum stellen die Funktionalitäten der Kollaborationswerkzeuge dem Anwender auf der Client-Seite zur Verfügung. Somit kann der Anwender auf die heterogenen Werkzeuge über die homogene Oberfläche des entstehenden wmc²-Portals zugreifen.

2.3 UMSETZUNGEN

Um die beschriebene Architektur auf ihre Tauglichkeit überprüfen zu können, wurde anhand eines Beispielszenarios prototypische Software entwickelt. Das Testszenario beinhaltet einen Chat und einen darin integrierten Datenaustausch (FileExplorer). [1] Im ersten Schritt wurden die dafür notwendigen Webservices implementiert, die auf die Funktionen von BSCW und Relax zugreifen. Danach entstanden zwei prototypische Webclients und zwei native mobile Prototypen.

3 USER INTERFACE DESIGN

Für die Erstellung der homogenen Oberfläche für das wmc²-Portal wird in erster Instanz der Fokus auf die Anwendung im Browser, also den Web-Client, gelegt. Hier können alle Funktionen bereitgestellt werden, wohingegen bei mobilen Interfaces der Funktionsumfang reduziert werden muss.
Das bisherige Designkonzept (siehe Abbildung 2) sieht eine Anlehnung an die Desktop-Metapher vor, bei der eine Leiste mit Funktionen zur Verfügung steht. Jede Funktion öffnet ein neues Fenster. Die Fenster können wie bei den gängigen Betriebssystemen verschoben, überlagert und geschlossen werden. Aktuell

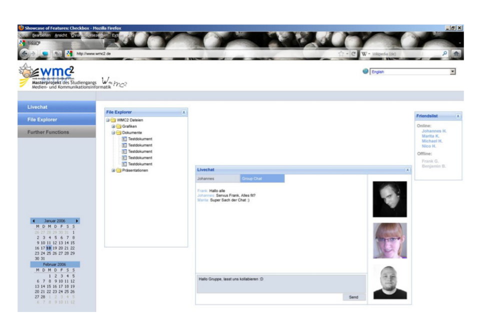


Abbildung 2: Aktuelles Designkonzept des Web-Clients

kann jedoch eine Veränderung der Desktop-Darstellung bei Betriebssystemen beobachtet werden.

3.1 VERÄNDERTES NUTZERVERHALTEN

Offenbar hat die verstärkte Nutzung von Smartphones die Entwickler von Betriebssystemen dazu angeregt, die Desktop-Metapher zu überdenken. Sie werden den Erwartungen der Nutzer von mobilen Endgeräten nun damit gerecht, die Darstellung der Funktionen von Smartphones und Tablet-PCs auch auf Desktop-PCs und Laptops zu übertragen. Apple hat bereits 2011 mit der neuen Version „Lion” des Betriebssystems OS X reagiert, bei dem beide Darstellungen möglich sind. Neben der gewohnten Desktop-Metapher gibt es nun das Launchpad, das vorhandene Apps wie auf dem iPhone darstellt (siehe Abbildung 3).
Auch Microsoft reagiert auf das veränderte Nutzerverhalten mit der neuen Windows 8-Version. Der Desktop gleicht nun der Oberfläche des WindowsPhones (siehe Abbildung 4). Durch



Abbildung 3: Launchpad von OS X Lion [2]

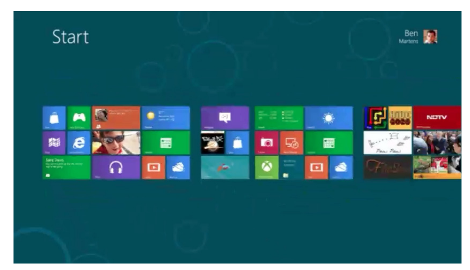


Abbildung 4: Desktop von Windows 8 [3]

die Beobachtung der sich ändernden Desktop-Darstellung stellt sich die Frage, ob die ausgewählte Desktop-Metapher des aktuellen Designkonzepts eine optimale und zukunftsgerechte Lösung für das Interface eines Kommunikations- und Kollaborationswerkzeuges ist.

3.2 ANFORDERUNGEN

Bei der Konzeption der neuen Oberfläche für das Kollaborationsportal wmc² sind zwei wesentliche Aspekte zu beachten. Der erste Aspekt liegt im modularen Aufbau des Systems, mit dem eine einfache Erweiterbarkeit gewährleistet ist. Diese Flexibilität muss sich auch im Interfacedesign niederschlagen, um zukünftige Funktionen integrieren zu können. Der zweite Aspekt zieht den Nutzer bei der Interface-Entwicklung mit ein. Damit die Kommunikations- und Kollaborationsprozesse optimal auf das Nutzerverhalten abgestimmt sind, werden Methoden des „User Centered Designs" [4] und des „Usability Engineerings" [5] S. 20-21, im Entwicklungsprozess angewandt.

3.3 VORGEHENSWEISE

Zu Beginn sollen die Stärken des aktuellen Designs herausgearbeitet werden, um sie für den weiteren Verlauf der Interfacegestaltung einbeziehen zu können. Es werden Zielgruppen bestimmt und Szenarien entworfen, die den Ablauf einer handelnden Person durch die entstehende Anwendung beschreiben soll. Parallel dazu wird der Funktionsumfang festgelegt. Der Funktionsumfang sowie die Szenarien werden für die Erstellung erster Prototypen genutzt. Die Prototypen werden phasenweise durch Usability-Tests evaluiert. Je nach Zielsetzung einer bestimmten Funktion werden die messbaren Faktoren [6], S. 27-28, unterschiedlich gewertet. So ist zum Beispiel bei der Aufgabe ein Dokument zu verschicken der Faktor Geschwindigkeit höher zu werten als der Faktor subjektive Befriedigung des Benutzers.
Die Tests und damit verbundenen Nutzerbefragungen sollen zudem Aufschluss über folgende Fragen geben:

- Wie kann ein interaktives System den Nutzer zur Kommunikation und Kollaboration motivieren?
- Wie verändert die Nutzung von mobilen Geräten das Nutzerverhalten an Laptops und Desktop-PCs?

4 VIRTUELLE 3D-WELTEN

Da bislang im Projekt wmc² die Visualisierung von Kollaboration nur auf 2D-Ebenen betrachtet wurde, soll sie in einer weiteren Arbeit in virtuellen 3D-Welten untersucht werden.
Diese virtuellen Computer-Welten sind laut [7], typischerweise: "*large social space where people network, build, play, buy and sell products, and work*" [7], S. 68.
Ein Vertreter solch einer 3D-Welt ist z.B. ‚Second Life' [7]. Dabei wird eine virtuelle Welt nicht nur verwendet um Menschen kennenzulernen, sondern auch um mit ihnen zusammenzuarbeiten. So wurde bei der Entwicklung der Funktionen von Second Life auf e-Commerce und e-Collaboration geachtet [8]. Aus diesem Grund können Firmen, wie z.B. Air France-KLM, diese Plattformen für Meetings verwenden. Auch verschiedene Bildungseinrichtungen, wie beispielsweise die University of Texas, verwenden virtuelle Welten um Trainings- und Lernveranstaltungen zu arrangieren [9]. Als weiteres Beispiel für die Verwendung virtueller Welten im Zuge von Bildungsaufträgen ist Sloodle zu nennen. Diese Software stellt eine Erweiterung für Second Life dar, um auf die Funktionen der e-Learning-Plattform Moodle zugreifen zu können [10]. Demgemäß zeigt sich, dass 3D-Welten bereits für verschiedene Arten von Kollaborationen genutzt werden.

Für die Untersuchung von Kollaborationen auf 3D-Ebene werden mögliche Szenarien erarbeitet und getestet, in denen eine Zusammenarbeit in einer virtuellen 3D-Welt stattfindet, die über eine herkömmliche Konferenz hinausgehen soll. Ein mögliches Szenario ist z.B. die Bearbeitung und Darstellung von Dateien und deren Ordnerstrukturen in einer virtuellen Welt.
Um dieses Szenario umsetzen zu können, wird an einer Testumgebung gearbeitet. In ihr sollen sich die Benutzer mit Hilfe ihrer Avatare bewegen und mit anderen Besuchern über die wmc²-Webservices zusammenarbeiten. Um die Vision eines modularen Systems von wmc² verwenden zu können, wird der bereits implementierte BSCW-Webservice eingesetzt, der auf den BSCW-FileExplorer und somit auf die Dateien zugreift.
Die Testumgebung wird mit der Open-Source-Software „OpenSim" gestaltet. Diese ist kompatibel mit der SecondLife-Client-Software (dem SL-Viewer) und kann analog zu SecondLife betrieben werden [11]. Für die Kommunikation über die Webservices werden Skripte in den Sprachen C# oder LSL (Linden Scripting Language) benötigt, die in der Testumgebung über virtuelle 3D-Objekte angewendet werden [12].
Die Verwendung von OpenSim hat mehrere Gründe. Es stellt zum einen ein Open source-Programm dar, das eine flexible und kostenlose Entwicklung der Testumgebung auf den eigenen Servern ermöglicht und die Software so vollständig an die Bedürfnisse des wmc²-Projektes angepasst werden kann [13]. Zum anderen simuliert OpenSim fast alle Funktionen von SecondLife [12]. Auf diese Weise erhält der Benutzer eine Plattform, mit der 3D-Welten und Avatare erzeugt und verwaltet werden können. Es ermöglicht zudem eine native Kommunikation der Avatare via Chat, Audioausgabe bzw. -eingabe und vordefinierten Gesten.
Wie bereits erwähnt, sollen die in einer Kollaboration entstehenden Dateien, wie z.B. Dokumente, Bilder usw., in einer virtuellen 3D-Welt dargestellt und bearbeitet werden.

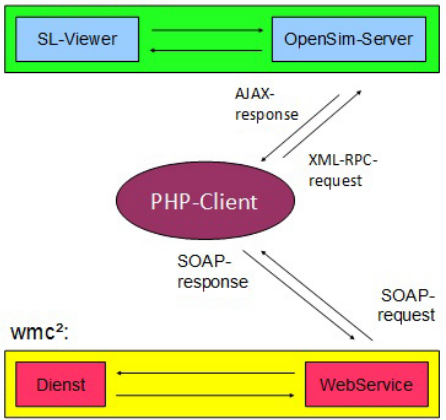


Abbildung 5: Kommunikation zwischen OpenSim und Webservices

Damit über OpenSim auf den BSCW-Webservice zugegriffen werden kann, muss ein weiterer Client implementiert werden, der die Anfragen von einem OpenSim-Objekt empfängt und an den Webservice weiterleitet, wie in Abbildung 5 zu sehen ist. Das OpenSim-Objekt versendet die Anfragen dabei als AJAX-Response. Der Client verschickt wiederum die Request des Webservices an das fragende Objekt über eine XML-RPC-Response zurück. Diese Vorgehensweise wird umgesetzt, weil ein Objekt in OpenSim nur über einen XML-RPC-Channel empfängt und nicht direkt über RPCs auf einen Webservice zugreifen kann [14].
Eine Frage, die bei dieser Arbeit beantwortet werden soll, ist, wie Dateien in einer virtuellen 3D-Welt dargestellt werden können. Eine mögliche Metapher für eine Ordnerstruktur stellt ein Schrank mit Schubladen oder ineinander liegenden Kisten dar, ähnlich einer Matrjoschka-Puppe. Über die Metapher der ineinander liegenden Kisten könnte rekursiv die gewünschte Navigationstiefe erreicht werden.
Eine offene Frage, die noch beantwortet werden muss, ist wie der Avatar mit diesen Dateien interagiert. Die Interaktion muss dabei das Bearbeiten von Dateien und die Manipulation von Ordnerstrukturen umfassen. Zu

diesem Zweck müssen auch Zugriffsrechte in der virtuellen Welt visualisiert werden.

5 AUSBLICK

Die Entwicklung neuer Kollaborationskonzepte für die Zukunft ist ein interdisziplinäres Forschungsfeld. Das Projekt wmc² wird nicht nur die Kollaboration im Hochschulkontext betrachten, sondern diesen auch auf den unternehmerischen und medizinischen Bereich ausweiten. Die Konzepte werden Geräte jeder Art, neue Interaktionstechniken und neue Funktechnologien mit einbeziehen.

6 LITERATURVERZEICHNIS

[1] Blubacher, Benjamin; Hornung, Nico: WMC2: Web*Mobile*Content*Collaboration, in: Informatics Inside; Kloos, Martínez, Tullius (Hrsg.); Reutlingen: Hochschule Reutlingen, 2011, ISBN 978-3-00-034591-3.

[2] https://www.apple.com/de/macosx/whats-new/launchpad.html, Zugriff: 11. März 2012.

[3] http://windows.microsoft.com/de-DE/windows-8/consumer-preview, Zugriff: 11. März 2012.

[4] http://www.usability.de/services/user-centered-design.html, Zugriff: 11. März 2012.

[5] Dachselt, Raimund; Preim, Bernhard: Interaktive Systeme; Berlin, Heidelberg: Springer-Verlag, 2. Auflage, 2010; ISBN 978-3-642-05401-3.

[6] Shneiderman, Ben: User interface Design; aus dem Amerikanischen von Jürgen Dubau und Arne Willner; Bonn: mitp-Verlag, 3.Auflage, 2002; ISBN 3-8266-0753-8.

[7] Jarmon, Leslie; Sanchez, Joe. 2008. Journal Of The Research Center For Educational Technology, Volume 4, Number 2. Gordon J. Murray, PhD; Kent State University School of Journalism and Mass Communication; USA

[8] Kock, Ned. 2008. International Journal of e-Collaboration. Texas A&M International University; USA

[9] GIGAOM. 2012. http://gigaom.com/collaboration/virtual-environments-for-training-collaboration-and-meetings/, Letzter Zugriff 11.03.2012.

[10] Sloodle. 2012. http://www.sloodle.org/moodle/, Letzter Zugriff 11.03.2012.

[11] OpenSimulator. 2012. http://opensimulator.org/wiki/Main_Page/de, Letzter Zugriff 02.03.2012.

[12] OpenSimulator. 2012. Scripting-Docu. http://opensimulator.org/wiki/Scripting_Documentation, Letzter Zugriff 02.03.2012.

[13] Weber, Steven. 2004. The Success Of Open Source. President and Fellows of Harvard College.

[14] Second Life Wiki. 2012. http://wiki.secondlife.com/wiki/Category:LSL_XML-RPC, Letzter Zugriff 20.02.2012.



E-Mail: infoinside@reutlingen-university.de

Internet: http://www.infoinside.reutlingen-university.de

Hochschule Reutlingen

Fakultät Informatik

Medien- und Kommunikationsinformatik

Alteburgstraße 150

D-72762 Reutlingen

http://www.mki.reutlingen-university.de

Telefon: +49 7121 271 - 4002

Telefax: +49 7121 271 - 4042





INF

Fakultät
Informatik



INF

Studiengang
Medien- und
Kommunikationsinformatik


ISBN 978-3-00-037938-3

9 783000 379383 >